FRACTURES FOR CLARINET AND COMPUTER

Gregory Hart Dixon, B.M., M.M.

Dissertation Prepared for the Degree of

DOCTOR OF PHILOSOPHY

UNIVERSITY OF NORTH TEXAS

May 2012

APPROVED:

Jon C. Nelson, Major Professor
Stephen Slottow, Committee Member
Andrew May, Committee Member
Lynn Eustis, Director of Graduate Studies in Music
James D. Meernik, Acting Dean of the Toulouse Graduate School

*Fractures for Clarinet and Computer* is a piece for live interactive performance using custom software designed in Max/MSP. The work explores musical borrowing and transformation of music from works such as Tchaikovsky’s Fifth Symphony, Gershwin’s *Rhapsody in Blue*, and several fragments from synthesizer recordings of the late 1960s and early 1970s. The dissertation focuses on both the musical aesthetics that informed the creation of the work and the software programming that enables live sampling and harmonization systems as well as flexible control of global parameters.
Copyright 2012

by

Gregory Hart Dixon
ACKNOWLEDGEMENTS

Thank you to Jon Christopher Nelson for his continuous guidance, support, and insight throughout the completion of this project and my studies at the University of North Texas. Thanks to my committee members, Andrew May and Stephen Slottow, for their invaluable expertise and assistance. I am also grateful to Joseph Klein, my academic advisor at the University of North Texas, and my former computer music mentors, David Bithell, Keith Kothman, Michael Pounds, Jody Nagel, and Cleve Scott.

Thank you to my family for their encouragement of my academic and musical pursuits. Thanks to clarinetist Rachel Yoder, to whom Fractures is dedicated, for her wisdom, patience, imagination, musicality, and collaborative spirit that have greatly influenced the composition of this work.
# TABLE OF CONTENTS

AKNOWLEDGEMENTS ...................................................................................................................... iii

LIST OF EXAMPLES ......................................................................................................................... vi

PART I: CRITICAL ESSAY .................................................................................................................. 1

Chapters

1. Introduction and Overview .............................................................................................................. 2

2. Musical Influences .......................................................................................................................... 3

3. Musical Materials .................................................................................................................. 10
   - Introduction .......................................................................................................................... 10
   - Sampled Recordings and Their Transformation .................................................................... 10
   - Score Quotations and Their Transformation ..................................................................... 13
   - Sampled Sounds, Instruments, and Hardware .................................................................... 17
   - Diversity of Acoustic Spaces .............................................................................................. 21
   - Diversity of Transducers for Sound Production and Recording .................................... 22

4. Form ............................................................................................................................................. 26
   - Overall Formal Design ....................................................................................................... 26
   - Gamut of Discontinuity and Continuity ............................................................................. 27

5. Interactivity and the Musical Roles of Performer and Computer ........................................... 29
   - The Computer as Accompanist and Equal Partner ............................................................ 29
   - The Role of the Performer in Relation to the Computer ..................................................... 30

6. The Use Of Improvisation In *Fractures* .................................................................................. 32

7. Software Development And Sound Design .............................................................................. 36
   - Physical Modeling with Waveguide Synthesis ..................................................................... 36
   - Max/MSP Real-time Sound Processing Algorithms ........................................................... 37
Real-time Polyphonic Recorder/ Sampler ............................................................... 38
Buffer Building ...................................................................................................... 39
Buffer Recording and Playback ............................................................................. 41
Polyphonic Voice Allocation ................................................................................ 49
Harmonization Algorithms ................................................................................... 53
Structural Design of the Main Performance Program ...................................... 57
Graphical User Interface Design of the Main Performance Program ............. 60
Conclusion ........................................................................................................... 65

Selected Bibliography ............................................................................................ 69

PART II: FRACTURES FOR CLARINET AND COMPUTER ..................................... 74

Program Notes ...................................................................................................... 76
Performance Notes ................................................................................................. 76
Technical Notes ..................................................................................................... 76

Fractures for Clarinet and Computer ..................................................................... 77
LIST OF EXAMPLES

Example 3.1: Fragment from Mort Garson’s *Exorcism* ................................................................. 12
Example 3.2 Tchaikovsky, Fifth Symphony in E Minor mm. 4-8 ...................................................... 14
Example 3.3: Opening Yamaha PSS-150 melody for computer (mm. 1-4) ........................................... 15
Example 3.4: The opening solo clarinet part for Gershwin’s *Rhapsody in Blue* (mm. 1-8) ... 16
Example 3.5: *Fractures* transformation of the opening clarinet solo from Gershwin’s

*Rhapsody in Blue.* ................................................................................................................................. 16
Example 3.6: A generalized studio diagram utilizing a variety of different types of

transduction .................................................................................................................................................. 24
Example 4.1: Formal Diagram of *Fractures* with Relative Durations of Sections ....................... 26
Example 6.1: Structured Improvisation (m. 25) .................................................................................... 32
Example 7.1: *buffer_build* Abstraction ............................................................................................. 40
Example 7.2: The First Instance of *Buffer_in_poly* Embedded Within poly~ .......................... 41
Example 7.3: Global View of *Fractures* Recording Program ............................................................. 42
Example 7.4: *buf_play_safe* Program ................................................................................................. 43
Example 7.5: *gd_event_gather* Program ............................................................................................ 44
Example 7.6: Duration Data Stored in the Collection *gd_events* ..................................................... 45
Example 7.7: Graphic User Interface for Sampler Control .................................................................. 46
Example 7.8: *stop_gesture* Program for List-based Control of the Sampler ................................. 48
Example 7.9: *gd_playback_controller* Program ............................................................................... 49
Example 7.10: Buffer and Voice Values Sent for Sample Playback .................................................. 50
Example 7.11: *Play_poly* Algorithm for Sampler Voices ................................................................. 51
Example 7.12: *event_mangler* Algorithm for Modifying List Data ............................................... 52
Example 7.13: The `smart_harmonizer` Algorithm................................................................. 53
Example 7.14: The `Ifo` Patcher .......................................................................................... 56
Example 7.15: Qlist Commands for Cues 11-15.................................................................. 59
Example 7.16: Graphical User Interface of the Main Performance Program ...................... 61
Example 7.17: `rehearsal` Patcher....................................................................................... 63
Example 7.18: `mock_clarinet` Patcher.................................................................................. 63
PART I

CRITICAL ESSAY
Chapter 1

Introduction And Overview

The term “electronic music” can suggest an incredibly wide range of sonic territory. *Fractures for Clarinet and Computer* highlights musical trends in the advent of commercial synthesizer music from the late 1960s and early 1970s and mixes them with contemporary classical electronic music techniques to create a work that embodies a multiplicity of musical styles.

*Fractures* features quotations from scores and recordings from the past that are transformed, destroyed, reconfigured, or distorted. These borrowed transformations represent the human emotion of nostalgia and how memories can feel transformed, hazy, and chaotic.

*Fractures* is intended to be a reflective, dark, somber, and otherworldly composition that strives to express emotional sorrow, bleakness, and pain. It also explores elusive beautiful aspects that can emerge from these dark feelings. The work is partially inspired by metaphorical ideas related to both exorcism and catharsis. I aimed to musically express catharsis and its extreme change in emotions by building up musical climaxes with manic, discontinuous, disorienting, and aleatoric music and contrasting these materials with reflective, continuous, and meditative music.
Chapter 2

Musical Influences

Electronic music and its tools have long shared commonalities with the tools and techniques utilized in the creation of popular music. This was especially true during early years of electronic music. For example, early electronic pop works such as Pierre Henry’s *Psyché Rock* (1964) and Raymond Scott’s *Soothing Sounds for Baby* (1962) were created with state-of-the-art electronic music tools closely related to the contemporary academic electronic works of their time. Indeed, Henry was an academic composer and pioneer of *musique concrète*. In an article about Pierre Henry in *Vital* magazine the author wrote, “In 1964 Henry produced his *Jerks Electronique* with a 'song' called *Psyche Rock* under the pseudonym Yper Sound. It sold some 150,000 copies. It made Henry instantly famous, not only with connoisseurs of avant garde art but also with the man in the street.” *Fractures* is a piece of interactive electroacoustic music that uses quotations from several distinct, early, popular electronic works to evoke a sense of nostalgia, longing for an era when electronic sound was still a brand new phenomena.

The use of commercially produced analog synthesizers by Robert Moog and Donald Buchla during the late 1960s saw a transformation from being used solely by avant-garde composers to frequent utilization by popular musicians. These instruments (especially Moog’s designs) went from being highly exclusive, purchased only by academic institutions

---

2 Raymond Scott, *Soothing Sounds for Baby Volume 1: 1 to 6 Months*, Epic LN 24083, 1962, LP.
or commercial advertising composers, to suddenly becoming more affordable and used commonly by keyboard players in rock n’ roll bands. The proliferation of the synthesizer was also largely a result of the enormous commercial success of *Switched on Bach* (1968) by Wendy Carlos, which introduced the Moog synthesizer and electronic music for the very first time to the general public. Carlos’ album proved that electronic music could be appealing to a wide audience. Trevor Pinch and Frank Trocco wrote, “*Switched on Bach* changed the face of pop, rock, and classical music- the first classical recording ever to go Platinum.”

Since the mid twentieth-century, several academic electronic music composers have borrowed inspiration or source materials from popular music. In addition, the availability of digital media offers possibilities for completely new approaches and techniques for musical borrowing. While musical borrowing has been prevalent throughout history in notated works of the past, electronic means make it possible to directly sample and transform not just the notation, but also the recorded sound. The composer now can place the original sound in a different context with entirely new surroundings. Simon Emmerson writes:

Sampling is strange. But what’s stranger still is how quickly we got used to it. I don’t just mean the way that, within a few years of it becoming widespread in the mid-eighties, it was hard to find a musician who objected on principle to having their music sampled. What’s really remarkable is how it just became an everyday

---

5 Wendy Carlos, *Switched on Bach*, Columbia Records MS 7194, 1968, LP.
part of our listening lives to enjoy – and to accept as music – records that were made out of fragments of other records: chunks of performance severed from their original time-and-place.\textsuperscript{7}

Sampling of popular music is now a pervasive technique found in many different electronic composers’ works.

Early examples of electronic works incorporating popular music include John Cage’s pieces utilizing radios, such as \textit{Imaginary Landscape No. 4} (1951) and \textit{Radio Music} (1956). Cage’s pieces are unique, because their incorporation of popular music is aleatoric, unintentional, and dependent upon the geographical location and sonic culture of its radio waves. These works, by accepting all manner of source material, unintentionally incorporate popular music if the performer playing the radio sweeps the radio’s frequency past a popular music station.\textsuperscript{8} In addition, Cage’s \textit{Imaginary Landscape No. 5} (1952) is a magnetic tape collage written to include any forty-two phonograph records, which could include popular music recordings if one desired. James Tenney’s electronic work, \textit{Collage No. 1 (Blue Suede)} (1961), deconstructed recordings of Elvis Presley.\textsuperscript{9} Stockhausen’s \textit{Hymnen} (1966-7) utilized recordings of national anthems throughout the world. Bernard Parmegiani wrote several works that utilize sampled recordings of rock, classical, and jazz combined with electroacoustic sounds such as \textit{Pop’eclectic} (1969), \textit{Du Pop à l’âne} (1969) and \textit{Jazzex} (1966). Mario Lavista’s tape work, \textit{Contrapunto} (1972), combines samples from various contexts and cultures creating a cultural superimposition or counterpoint.

Lavista explained the significance of this cultural interplay in his program notes for *Contrapunto*:

The elements that conform the musical material of this work have a diverse and maybe contradictory origin. Integrating these elements in a coherent structure implies difficult problems for the formal aspect of the piece, especially owing to the heterogeneity of the materials. Mahler’s 9th symphony, The Rolling Stones, music from the Japanese imperial court named *gagaku*, fragments of *noh* theater, electronic sounds, organ, The Beatles, a waltz by Richard Strauss, Zen Buddhism music, a USA radio station, national anthems from México, Japan and USA, etc., are all elements that appear in the work, sometimes literally. All of them have been manipulated and transformed until becoming unrecognizable at times, due to the electroacoustic techniques used such as filtering, frequency and amplitude modulation, pitch variation and editing. This is why in some passages of the work electronic sounds can be confused with a fragment of Mahler’s 9th, or Buddhist chants with Jimmy Hendrix for example. The superposition of such elements and their articulation in time originate a polyphonic texture and counterpoint interplay.\(^\text{10}\)

As Lavista notes, utilizing heterogenous musical materials presents distinct challenges to creating a unified formal structure for a work. *Fractures* also grapples with the challenges of utilizing heterogenous materials to create a unified formal structure (as discussed in Chapters 3 and 4) with varied approaches to synthesis and sound processing (as discussed

in Chapter 5). This survey of works represents of the earliest pieces of electronic music that sample or borrow musical material from popular music.

Newer electronic or electroacoustic works by composers that borrow inspiration from popular music include “plunderphonic”\textsuperscript{11} works by John Oswald, such as \textit{Plunderphonics LP} (1988),\textsuperscript{12} which samples popular recordings by artists such as Michael Jackson and Dolly Parton. Oswald’s album, \textit{Plexure} (1993),\textsuperscript{13} samples just over one thousand different pieces of recorded popular music.\textsuperscript{14} Jim O’Rourke’s 10” vinyl recording, \textit{Please Note Our Failure} (1998),\textsuperscript{15} utilizes various samples from popular music recordings. Andrew May’s \textit{Still Angry} (2007) for flute, clarinet, and computer, utilizes fragments of cover versions (performed by May) of songs by punk bands from Manchester, England during the 1970’s. May also transforms these quotations into new notated forms for the clarinetist and flutist.\textsuperscript{16} Ricardo Climent’s fixed media work, \textit{Sir George} (2008), takes its inspiration from analog recordings made during the 1960’s and 70’s by the producer and “fifth-Beatle,” George Martin.\textsuperscript{17} Bob Ostertag created the electronic work \textit{w00t} (2007),\textsuperscript{18} which borrows its samples solely from popular video games. This survey of examples demonstrates a longstanding interest in utilizing quotations of popular music within the electroacoustic genre.

\begin{flushright}
\footnotesize
\textsuperscript{11} John Oswald, “Plunderphonics, or Audio Piracy as a Compositional Prerogative,” Plunderphonics, \url{http://www.plunderphonics.com/xhtml/xplunder.html} (accessed January 26, 2012)
\textsuperscript{12} John Oswald, \textit{Plunderphonics}, Plunderphonics Mystery Tape Laboratory WRC1-5744, 1988, LP.
\textsuperscript{13} John Oswald, \textit{Plexure}, Avant Avan 016, 1993, CD.
\textsuperscript{15} Jim O’Rourke, \textit{Please Note Our Failure}, Some some-08, 1998, 10” vinyl.
\textsuperscript{16} Andrew May, “Andrew May- Still Angry- recording and score,” \textit{Andrew May- Home Page}, University of North Texas, \url{http://cemi.music.unt.edu/may/Works/Still_Angry.html} (accessed January 26, 2012)
\end{flushright}
Simon Emmerson writes “Earlier eras had their own obsessions with antiquity, of course, from the Renaissance’s veneration of Roman and Greek classicism to the Gothic movement’s invocations of the medieval. But there has never been a society in human history so obsessed with the cultural artifacts of its own immediate past.” An exhaustive list of electronic works incorporating popular recorded media today is so broad that it would be practically impossible to compile.

In addition, the compositions and musical aesthetics of composer Luc Ferrari were an important influence on my composition. Ferrari’s compositions are concerned with intimacy, and he has discussed this intimacy in relation to his use of recordings of anecdotal spoken word in various languages. Ferrari said, “To discover speaking through the medium of recording is a rather astonishing thing. When I heard this natural speaking, I discovered its intimacy and its musicality, its manner of communicating sensitivity.” He later states that, “Speaking is so intimate. It comes from the deepest part of us: from both the head and the sexual organs, from the heart and from all that we can imagine.” While Fractures does not contain any recordings of spoken words, it is influenced by Ferrari’s notion of intimacy by incorporating the performer’s own improvisational musical language and the unique voice of the performer into the work through the instruction to scream through the clarinet.

Fractures is also influenced by the concept of tautology that is found in Ferrari’s series of Tautologos works. Ferrari states, “In these pieces I developed a manner of

---

21 Ibid.
repeating by deviation. The deviation prevents the repetition from becoming a new rule.”^{22} Ferrari’s concept of tautology in *Tautologos* combines the conflicting ideas of repetition found in minimalism with the strictly non-repetitive aesthetics of serialism.^{23} There are several portions of *Fractures* where the clarinetists is called to perform varied repetitions of a fundamental musical idea, such as the clarinetists insistent rising and falling material during what I call the “Mantra” section (Section B) and the unrelenting varied chromatically falling figures employed at the end of the work during Section G.

Ferrari’s utilization of improvising performers is another important influence for *Fractures*. The documentary *Facing His Tautology* (2005)\(^{24}\) displays Ferrari during the last days of his life creating a new recording of *Tautologos III*. In this fascinating documentary, we are shown how Ferrari produces a realization of the work for electronics, piano, and viola. He is working in the studio with two improvising musicians, pianist Jean-Philippe Collard-Neven, and violist Vincent Royer. *Tautologos III*’s score contains no pre-composed notations for the musicians. Instead, they are asked to improvise based on Ferrari’s distinct system of rules composed for the work. Ferrari’s role as composer becomes that of sonic overseer or guide who is sculpting the sound of the composition through editing, giving directions to the performers, and receiving feedback from their improvisations that inform the work’s realization as it progresses. This is closely related to my process in composing *Fractures*, since I am using the studio as a tool for sonic exploration and transcribing recordings of structured improvisations, which are incorporated into the larger work.

---

\(^{22}\) Ibid., 13.

\(^{23}\) Ibid., 13-4.

Chapter 3
Musical Materials

Introduction

Many diverse musical materials are utilized in the composition of Fractures. Though the material is primarily original, sections of the work employ very short samples of borrowed works along with altered forms of well-known works from the clarinet repertoire. The borrowed materials include samples from various long-playing vinyl records (LPs) and transformations of other composers’ scores into newly notated musical materials. Borrowed samples and quotations are transformed further by a variety of musical hardware and computer processing techniques. As a result, only a distorted allusion to their original context remains.

Other samples that contribute to the fixed media component are my own recordings of Rachel Yoder playing clarinet and bass clarinet. In addition, Fractures uses samples that provide contrast with the clarinet. These samples include recordings of a cuíca, metal pan lids, a rolling clarinet bell on wooden and metal surfaces, and the sounds of cars and trucks driving by a factory.

Sampled Recordings and Their Transformation

While composing Fractures, I recorded many samples from LPs that featured synthesizers or organs from lesser-known artists of the 1960’s and 70’s. These obscure recordings collectively have a unique sonic fingerprint. Limiting the cultural exposure, and general time period of the sampled recordings is intended to give the piece a more nostalgic and personal identity.
I sampled very short fragments of LP’s such as Dick Hyman’s *The Electric Eclectics of Dick Hyman* (1968),
25 Ruth White’s *Short Circuits* (1971),
26 Bob Hacker’s *One Man Opry: Bob Hacker Plays the Yamaha Electone D-80* (year of publication unknown),
27 and Wendy Carlos’ *Switched on Bach* (1968)
28 and several others. These records all managed to capture distinct clarinet-like timbres, through exposed passages usually originating from the artist’s square-wave generators. These were the passages I usually chose to sample, because their clarinet-like timbres blended well with the sound of the live performer. Aside from electronic representations of the clarinet, I also utilized very brief samples of acoustic clarinet multiphonics played by William O. Smith featured in his composition, *Fancies for Clarinet Alone* (1969),
30 Perhaps the most exceptional, clearly transformed, borrowed quotation from *Fractures* is a fragment from the composition, *Exorcism*, by Mort Garson from his album, *Black Mass* (1971, released under his pseudonym, “Lucifer”).
31 The first three notes of the melody outline a chromatic descent, a prevalent melodic archetype utilized throughout *Fractures* (Example 3.1).

\[\text{\footnotesize 26 Ruth White, } \textit{Short Circuits}, \textit{Angel Records} S-36042, 1971, \textit{LP}.\]
\[\text{\footnotesize 27 Bob Hacker, } \textit{One Man Opry: Bob Hacker Plays the Yamaha Electone D-80}, \textit{Yamaha, YR-5007}, \textit{LP}.\]
\[\text{\footnotesize 28 Wendy Carlos, } \textit{Switched on Bach}, \textit{Columbia Records, MS} 7194, 1968, \textit{LP}.\]
\[\text{\footnotesize 29 William O. Smith, } \textit{Fancies for Clarinet Alone}, \textit{(New York: MJQ Music, 1972)}.\]
Example 3.1: Fragment from Mort Garson’s *Exorcism*

This selection also features an interesting melody in the middle voice closely related to the inversion of the upper voice. After the initial presentation, the first two measures are repeated transposed down a major second.

This sample is found in an extended passage near the middle of the original song. Rather than use a lengthy audio sample, I decided to recreate this borrowed material by transcribing the material from the LP for clarinets and bass clarinet and then made multi-track recordings of the different parts with the clarinetist. These recordings are then transformed further through studio sound processing.

In the sixth section of *Fractures* (Section F), samples from old recordings are the most prevalent. The electronic music’s texture here is created through a technique known as concatenative sound synthesis or sonic mosaicing. Concatenative sound synthesis is made with a group or corpus of sound samples that each collectively can become a part of the timbral properties of the subsequently generated sound. Concatenative synthesis is like granular synthesis in that it usually works with smaller fragments of sounds. Concatenative synthesis is distinctively different than granular synthesis in that it first analyzes the sound fragments from the corpus and determines specific sonic descriptors about each fragment, such as pitch, loudness, periodicity, spectral flatness, spectral
centroid, and several other parameters. Concatenative synthesis then allows the user to combine fragments of the corpus that have similar characteristics (or not) and synthesize a sound that potentially consists of sonic fragments from either the entire corpus of sound files or a small subset of files. The disparate samples from the corpus and their summation create new and unique synthesized musical textures that also retain a timbral relationship to the original sounds.

Concatenative synthesis was a particularly attractive processing method for Fractures because it suggests the possibility of integrating a diverse and wide gamut of sounds into one. Given the use of numerous and divergent sound samples in Fractures, concatenative synthesis offered a helpful way for me to unify, integrate, or meld these heterogeneous materials together. It also provided a way of algorithmically creating textures from a wide array of sounds without necessitating extensive splicing and mixing in a digital audio workstation’s multi-track sequencer. In addition, I performed concatenative synthesis on smaller groups of sounds that were more homogeneous; such as a corpus of samples of the clarinetist screaming through the clarinet while playing different specified pitches.

Score Quotations and Their Transformation

Fractures uses fragments from several seminal compositions for clarinet, though their presentation in the score for the clarinetist is always altered or distorted from the original. While considering numerous masterpieces of clarinet literature, I selected the famous opening passage of Gershwin’s Rhapsody in Blue (1924), and the solemn, insistent, descending melody in the opening of Tchaikovsky’s Fifth Symphony in E Minor (1888) for use in Fractures.
The quotation from Tchaikovsky consists of the clarinet passage found in the first movement, first introduced in measures 4-8. Tchaikovsky crafts the original melody with a beautiful economy of means since it is a simple, linear melody with just six notes of the E natural minor scale. Tchaikovsky gives this fragment of the melody interest and impact by utilizing harmonic variation in its accompaniment in the opening of the work (Example 3.2). This six-note fragment becomes a transformative musical tool for Tchaikovsky.

Example 3.2 Tchaikovsky, Fifth Symphony in E Minor mm. 4-8

Tchaikovsky’s motive is greatly concerned with linear stepwise motion in each of the voices, overlapping layers of upward and downward scalar movement. This concern with linear stepwise motion also proliferates the melodic materials of Fractures.

The first transformation of this material includes scoring Tchaikovsky’s clarinet part for synthesizer and electronics. The opening of Fractures states a variation of this melody played on the Yamaha PSS-150 synthesizer. The melody is transformed so that it continues steadily downward in E natural minor. A delay was then applied to the PSS-150’s signal so that the same melody echoes a beat and a half later. Feedback of the delay is also utilized so that more copies of the original line follow and continue to flow downwards. This delay time creates an interesting rhythmic syncopation and also an interesting canonic melodic pattern (Example 3.3).
Example 3.3: Opening Yamaha PSS-150 melody for computer (mm. 1-4)

Interestingly, I found that this transformation elicits association with the famous melody from the Latin hymn, *Dies Irae (Day of Wrath)*. This melody has been transformed by many other composers and is used to symbolize death.

In addition, I transformed the Tchaikovsky fragment in more abstract ways. The descending diatonic scalar pattern found in Tchaikovsky’s clarinet part is transformed into continually descending chromatic scales or fields. The descending chromatic scale is utilized in *Fractures* for its historic musical symbolism for grief and sadness. Chromatic fields of notes, because of their symmetrical nature, have a transparent sound quality creating a feeling of stasis, while still containing all twelve tones. The borrowed fragment from the Symphony in E minor also has a very rigid, transparent quality that I particularly admire.

Similarly, the clarinet passage from Gershwin’s *Rhapsody in Blue* is transformed in several different ways in *Fractures*. The Gershwin material first appears in the clarinet part during the first section of the work. The piece begins with a transformation of the clarinet’s opening line of *Rhapsody in Blue* (Example 3.4).
Example 3.4: The opening solo clarinet part for Gershwin’s *Rhapsody in Blue* (mm. 1-8)

Example 3.5 displays the transformation of the opening clarinet solo from Gershwin’s *Rhapsody in Blue* from the opening of *Fractures*.

Example 3.5: *Fractures* transformation of the opening clarinet solo from Gershwin’s *Rhapsody in Blue*.

This opening clarinet gesture of *Fractures* follows the relative rhythm and contour of the original. However, the transformation generally alters the pitch content of the original line to be more chromatic. The opening trill is a minor-second and the initial ascent is chromatic. This opening glissando ultimately arrives on C-sharp rather than C, subverting the dominant tonic relationship of the original and emphasizing a tritone. This is
harmonically significant because the original context supports a dominant to tonic relationship from the opening G to its ultimate goal, C. Additionally, the clarinetist is instructed to play this note and aggressively sing, yell, or scream, into the clarinet within a minor third of the notated pitch, hence the altered notation of the C-sharp.

While the opening chromatic scale still sounds comparable to the original part, when the clarinetist reaches the scream on the C-sharp, there is a startling shift signifying an entirely different work (the electronic music also helps the case). This juxtaposition of musical elements thwarts the listener’s expectations, assuming the listener is an informed listener who is familiar with Gershwin’s composition. The listener’s expectation is further thwarted by the presence of the electronic accompaniment that supports this C-sharp.

*Rhapsody in Blue’s* introductory clarinet solo is an explosive, exciting, highly recognizable introduction that is a musical model worthy of imitation. It exploits the clarinet’s abilities for glissandi, which are utilized throughout *Fractures*. In addition, I wanted the piece to start out with a slightly recognizable score transformation as a signpost for the listener that the work is interested in musical borrowing and transformation.

*Sampled Sounds, Instruments, and Hardware*

All the sounds that were sampled for the fixed media component of *Fractures* exhibit some relationship to the clarinet. Some of the electronic sampled sounds clearly emulate the clarinet, while others are related in an abstract or highly personal way. Much of the material was sampled from very short fragments of commercially available synthesizer LP recordings. All of the samples have timbres that are closely related to the sound of the
clarinet. This happens frequently when synthesizer players use square wave generators, which share the overtone relationship of odd partials with the clarinet.

Moreover, the samples selected were limited to those found on early commercial albums for synthesizers written during the late 60’s and early 70’s. These recordings were most attractive for use in my piece as a result of the artist’s general desire for musical innovation and distinct timbres using novel instruments. These recordings were made when the sound of the synthesizer was first being expressed and only played by a select few artists. Often, I find these early synthesizer recordings display a great deal of charm.

Many of the instruments that the early synthesizer artists were using were also rare and often even one-of-a-kind. Consequently, they are hard to find, expensive (if they are even still operating), and playing and recording them could have proven to be very difficult. Their albums provide an interesting and broad range of sonic material that can be explored further.

When one listens to the early synthesizer albums by Wendy Carlos and her contemporaries, the corporeal element is at the forefront.32 These older synthesizer artists realized their sounds in the studio by relying almost completely upon the physical touch of the keyboard. I feel that the aspect of corporeality of a musical sample is a quality that can often be heard and I wanted the music for Fractures to have a corporeal or human element.

Some composers feared that this corporeal element was lost when digital sequencers became available. Carlos described her antipathy to the pre-composed features of digital sequencing in the liner notes of Patrick Gleeson’s album Beyond the Sun,

---


> We are looking forward to discussing with Gleeson and others at long last the issue of methodology: humanized live keyboard performance vs. automated sequencer-control of pitch and duration. We strongly favor the former approach while Gleeson now employs the latter. But while the accurately rigid translation of the written score via Gleeson’s system has undeniable merits, it does lead to a subtly mechanized feel to the music, and thus represents the only major area of disagreement between us.

Carlos’ insistence on retaining the corporeal element in electronic music is clearly seen by her disparaging musical view of Gleeson’s use of sequencers.

The virtual nature of many of the current tools utilized for electronic music composition lacks tactility. This is embodied in many of the software-based sequencers, plug-ins, and DAWs. Currently, this problem is being confronted in many ways as composers interface with devices such as mobile phones, iPods, and other portable technology. Many composers also work to develop new interfaces and physical computing hardware.

With *Fractures*, I was inspired to create electronic music made corporeally, often by human touch as an homage to the early synthesizer pioneers. Several keyboard synthesizers were played and recorded to generate musical materials for the work. These

---


34 Wendy Carlos (1976), In *Beyond the Sun, An Electronic Portrait of Holst’s The Planets, Patrick Gleeson, Eu Polyphonic Synthesizer* [LP back cover], Chicago: Mercury Records.
included the Yamaha TX-81Z, Kurzweil K2000, and the Yamaha PSS-150 that is heard at the opening of the piece. I also created a MIDI-controllable waveguide clarinet instrument in Csound that was played with a MIDI controller. I recorded several MIDI-capable synthesizers without using MIDI sequencers to synchronize the performances. In addition, I recorded samples generated using gestural finger motion on an iPod-using the Moog Filtatron.

The advent and proliferation of new touch-based technologies for electronic devices such as iPods, iPads, etc., offers an exciting new paradigm for creating electronic music guided by human touch or control. Portable devices have already started a new resurgence of physical, tactile-based electronic instruments. Fractures addresses these new trends and synthesizer design trends from the past by predominantly utilizing tactile instruments to generate fixed media. Almost all of these synthesizers utilized inherently needed to be grasped or touched in order to make their sound.

Signal processing was also subject to touch-based gestural control. I processed several sound sequences from the computer through an Electro-Harmonix Memory Man analog delay guitar pedal. This pedal allows the user to easily control variables like delay time, chorus or vibrato, wet/ dry mix and feedback, all by hand. To create a stereo delay effect with this monophonic device I recorded two different passes of each electronic sequence through the Memory Man. Each pass was recorded while manipulating the pedal’s knobs to create similar gestures and then I spread the two monophonic effect “sends” by panning them to the left and right; maximizing their separation and uniqueness.
Similarly, the concatenative synthesis for Fractures used gestural mouse movements to create timbral or textural shifts using the software “cataRT.”

Diversity of Acoustic Spaces

There are many different types of acoustic spaces explored in Fractures. Many acoustic spaces are already inherent in the different commercial sound recordings that I sampled, creating many opportunities for shifts in acoustic space. In addition, I recorded many sounds in my studio, kitchen, and in the outdoors that each provide a different sonic space.

For additional hardware effects processing, I played back sequenced sound from the digital audio workstation (DAW) into a re-amp. A re-amp is a device that converts the computer soundcard’s line level output signal into a signal with an input impedance matched for guitar amplifiers and equipment. From the reamp I then sent the computer’s signal through an Electro Harmonix Memory Man analog delay pedal, and finally through a vintage Versatone Pan-o-Flex guitar amplifier. I then recorded the computer audio played through the guitar amp at close range with a dynamic microphone. The Versatone has a spring reverb and vibrato effect that can be mixed to taste. This technique of re-amping pre-recorded sounds is closely related to the studio use of echo chambers. The unique coloration of the stompbox, guitar amp, microphone, and room are all being applied to the original source sound. This creates a new variety in transduction methods and also acoustic space. The Versatone tube amp, in combination with the Memory Man, provides

---

an analog warmth and subtle distortion that has a fidelity that is less-defined, hazy, or murky. This quality sounds similar to older synth recordings and vintage analog delay effects.

Sonic space was also altered in Fractures by processing sounds on the computer. In the recreation of the fragment of Mort Garson’s *Exorcism*, I recorded the clarinet parts and then processed them with filters, digital and analog delay, and reverb. The mixes are convoluted with impulse responses of old vintage amplifiers and speakers that Fokke van Saane has distributed.\(^{36}\) This creates audio that sounds like it was being played through vintage loudspeaker systems. Mixing these convolved samples with the original dry sample creates a continually changing sonic space.

*Diversity of Transducers for Sound Production and Recording*

Perfectly linear response through transduction is theoretically never possible. With Fractures, I wanted to highlight several different types of transduction, each with their own unique imperfections and sonic character. Electronic music can incorporate a multitude of disparate sound processes relying upon vastly different methods of input and output transduction. These include hardware amplifiers, magnetic pickups, stomp boxes, effects chains, speakers, microphones, spring reverbs, tape delay, echo chambers, and a number of other devices. Composers can utilize additional tools to assimilate various forms of transduction. With signal routing, these transduction methods can be easily translated or fed back into the signal chain if desired. Fractures explores several options of transduction and sound processing outside of the computer.

---

It can be difficult to model an older sound since the equipment today is dramatically different. Older recordings often have non-linear distortion that results from analog equipment such as vacuum tubes, magnetic tape, and transformers that today are less widely used. Fractures strives to attain this older sound though the use of convolution, the use of some analog equipment, and through direct sampling of older recordings.

Composer Ricardo Climent created a similar synthesis of transduction methodologies in his work *Sir George* (2008). Climent writes:

*Sir George* is a fixed media composition which serves as a point of encounter between legendary British pop-rock sound and transistorized contemporary computer music synthesis of today. The sonic experience is a journey riding on top of a turntable’s needle, rediscovering the sound flavour of tube-operated mixing consoles combined with classic outboard gear and four-track one-inch recorders and confronting it with a glitchy hi-tech computerized sonic scenario.\(^{37}\)

A sound engineer can achieve a consistency of sound by using the same microphone setup, A/D converters, preamps, sequencers, and DAWs. However, in electroacoustic music, consistency can sometimes lead to bland results from the urgent and insistent use of the same kinds of hardware tools and recording methodologies. The composer or sound artist has the opportunity to fantasize, distort, and willingly break or extend the rules governed by the common musical and studio production ideologies; creating new ways to represent sound, time, and space, and portray it to an audience in a unique and artistic way. *Fractures’* source material comes from a fractured, disparate assortment of studio sessions,

---

and varying forms of transduction which each translate their unique sonic fingerprint, characteristic, personality, or vibrancy. Example 3.6 shows a generalized diagram of how the music studio was configured in Fractures to achieve a variety of different types of transduction.

Example 3.6: A generalized studio diagram utilizing a variety of different types of transduction.

The fixed media component of Fractures often features material that has been colored by added sonic distortion(s) or artifacts. Unlike purely digital media, which can theoretically be perfectly replicated, sonic artifacts in fidelity are intrinsic from repeatedly
playing analog sources such as magnetic tape and LPs. Sonic color, artifacts, and their
distortions can contribute to a listener's sense of nostalgia and the past if they are familiar
with the distortion characteristics of those past technologies. When playing or sampling
LPs, the stylus, record, and phono amplifier all provide their own unique set of distortion
characteristics. When a phonograph’s signal hits the analog-to-digital converter for
sampling there are already several layers of sonic colorings or distortions that have been
applied through various transduction methods.

Transformation of sampled recordings and score quotations, along with utilization
of a diverse acoustic spaces and transducers for sound production and recording are
integral features to the fixed media component of Fractures. These musical materials are
woven into the work in a musical dialog with the live clarinet and its real-time effects
processing. These elements create a sense of musical heterogeneity and discontinuity that
is an important element in the work’s formal structure.
Chapter 4

Form

*Overall Formal Design*

*Fractures* is a through-composed rhapsodic work that contains seven main sections
(Example 4.1).

![Diagram of Fractures with Relative Durations of Sections]

Example 4.1: Formal Diagram of *Fractures* with Relative Durations of Sections

These seven sections (rehearsal marks A through G in the score) all share a similar musical
language, but often contrast distinctly with one another in musical timbre and texture. The
contrasting timbres are frequently characterized by either continuous (stable) or
discontinuous (unstable) dynamic morphologies.38 Trevor Wishart writes that sounds of
unstable morphology “flip back and forth between a number of distinct states” and “such

---

sounds are coherent in the sense that the overall field of possibilities remains constant but
the immediate state of the object is constantly changing in a discontinuous fashion.”
Discontinuous dynamic morphologies of sound are made possible through the combination
of wide varieties of recordings, non-linear editing techniques, granular or concatenative
synthesis, and radical shifts in musical space. Fractures transforms fragmentary materials
from other works in recorded form to create discontinuous sound morphologies. In
contrast, continuity in sound is achieved by limiting change in musical parameters such as
morphology, texture, acoustic space, rhythm, melody, or harmony. Three sections (C, E,
and G) are highlighted by musical materials with stable parameters, while four sections (A,
B, D, and F) feature materials with discontinuous parameters.

Gamut of Discontinuity and Continuity

There are important climactic moments in Fractures where a sharp contrast is made
between discontinuous and continuous music. These climaxes occur at the conclusions of
sections B, D, and E. The first two climaxes result from a gradual accumulation of materials
that becomes very intense at the conclusion of sections B and D. The arrival of more
continuous, simpler material at the start of sections C and E provides similar responses to
these climactic moments. At the arrival of section C, the sampler is quickly playing back
accumulated samples from the previous section. It gradually stops playing back samples
until its texture thins down to nothing. The next section continues with solely the sound of
the clarinet processed by real-time phaser and delay effects. This creates a dramatic sense
of contrast by shifting into a more subdued and homogeneous musical texture.

---

When section D arrives an accumulation of discontinuous music begins again and builds towards a climax at the end of section D. Following this accumulation of sound, the musical texture becomes simpler and much clearer with the arrival of the more continuous and harmonically stable “Exorcism” transformation found in the electronic music of section E. This release of energy only lasts for the short duration of section E and is obliterated by the immense rush of discontinuous energy and sound that characterizes the following section (F). Finally, the closing section (G) provides contrast to section F as it is highlighted by more stable rhythm, timbre, and dynamic morphologies.
Chapter 5

Interactivity And The Musical Roles Of Performer And Computer

The Computer as Accompanist and Equal Partner

The computer part of Fractures contains three main elements that complement the clarinetist’s performance: fixed media playback, real-time effects processing, and real-time sampling and playback. These elements are each independent and can often be operating simultaneously. They contribute to musical behaviors for the computer that range from accompanimental to soloistic.

The fixed media materials frequently accompany the performer by providing sonic ambience. These materials often lack stressed beat patterns or any kind of soloistic quality. As a result, they can be synchronized loosely with what the clarinetist is playing. However, other portions, such as section D, contain sections where at the foreground level the clarinetist and fixed media are playing parts that are equal, like a duet. Here, synchronization between the soloist and computer is required. The computer is featured during section E, where the clarinetist takes on a strictly accompanimental role.

The sampler also behaves both as an accompanist and equal partner to the clarinetist. It plays back recordings in counterpoint with the clarinetist to create duets (mm. 22-26). The computer software accumulates these live samples and eventually plays back faster sequences of short samples. In these spots (mm. 28-40 and mm. 67-75) the computer component loses its melodic character and instead creates a granular sound texture that is used to accompany the solo clarinetist.
The real-time effects processing in *Fractures* contributes as a musically equal partner that extends the clarinetists’ sound at times and also contributes as a soloist. The qualities of an equal partner are exemplified by the use of effects such as reverb, delay, harmonization, amplitude modulation, and phaser when used to directly process the clarinetist’s live sound. The real-time effects sometimes include a harmonizer with pitch modified by a low frequency oscillator (LFO). When the clarinetist holds out a single tone with this effect, a melodic line is created. In this instance, the clarinetist as instigator of the source sound is a very natural accompaniment to the melismatic material generated by the harmonizer. When the pitch and tone of the clarinet change over time, the harmonizer’s timbre follows suit. These harmonizer melodies are featured at the end of the work in sections F and G. Section F is characterized by harmonization featuring an LFO that is fast and includes a wide pitch range. In contrast, the harmonizer in section G uses an LFO with a frequency that slows down while the pitch range diminishes over time.

*The Role of the Performer in Relation to the Computer*

Some features of the computer audio are impossible to create without the input of the clarinetist. These include both real-time sound processing of the clarinet and live recording of the clarinet for sampler playback. The latter also requires the active participation of the computer technician who must follow the score to trigger audio recording at various points throughout the score. Conversely, the clarinetist receives numerous auditory cues from the computer. These cues include timing indications, rhythmic cues from the playback of fixed media, sample playback identification, and various effects.
Fractures provides an interactive musical environment where various computer-based technological elements interact with the performer. The interactivity and relationships between the performer and the computer together create a complex and highly flexible musical ensemble.
Chapter 6

The Use Of Improvisation In Fractures

Improvisation is an important performance element of Fractures and also plays an important role in the composition of the work. There are some improvisatory sections written in the score for the clarinetist in section B where a curvy line, along with the marking “ad lib.” outlines the general pitch trajectory or contour for the clarinetist. Example 5.1 shows an example of this sort of directly guided or structured improvisation. The pitch and especially the rhythm are indeterminate during these improvisatory sections of the work.

Example 6.1: Structured Improvisation (m. 25)

In addition, section B (Mantra) has an open form, and the clarinetist is asked to play the various systems during the section in any chosen order, improvising and creating the overall phrase structure during the performance of the work. This improvisation results from the use of a mobile score and provides more choice than the more structured forms of improvisation.

The clarinetist must make a plethora of musical decisions when improvising with the sampler. Throughout section B the sampler is playing back improvised material that the clarinetist has just played, at different speeds and on occasion backwards. At times, the
clarinetist also improvises with the computer’s transformations of her own improvised material. In this section, the clarinetist must be familiar with the processes that can result from this unusual improvisatory musical relationship.

During section B, the technician is also given the opportunity to improvise. *Fractures*’ sampler is often played with a technique that I call the stop gesture. A stop gesture is pressed after a recording of the clarinet is made and accomplishes two things: it stops the recording, and immediately triggers the sampler to play back what had just been recorded in one of several prescribed ways. The score indicates stop gestures, but the technician must choose which of the eight different types of stop gestures is selected in each instance. These stop gestures are as follows:

**Stop Gesture a:** immediately plays the prerecorded sample at normal speed. (played by depressing 1)

**Stop Gesture b:** immediately plays the prerecorded sample transposed up an octave. (played by depressing 2)

**Stop Gesture c** immediately plays the prerecorded sample transposed down an octave. (played by depressing 3)

**Stop Gesture d:** immediately plays the prerecorded sample transposed randomly within a two-octave span. (played by depressing 4)

**Stop Gesture e:** immediately plays the prerecorded sample backwards at normal speed. (played by depressing 5)

**Stop Gesture f:** immediately plays the prerecorded sample backwards transposed up an octave. (played by depressing 6)
**Stop Gesture g:** immediately plays the prerecorded sample backwards transposed down an octave. (played by depressing 7)

**Stop Gesture h:** immediately plays the prerecorded sample backwards transposed randomly within a two-octave span. (played by depressing 8)

During this section, surprisingly different results occur each time the performer and the technician both improvise.

In addition, portions of the notated clarinet part are taken from structured improvisations recorded during rehearsals and recording sessions. Some preliminary recording sessions provided minimal directives to the clarinetist. For example, the final section of the work (G) comes from a recording session where I asked clarinetist, Dr. Rachel Yoder, to play sorrowful material at the tempo of a dirge, mostly dominated by descending chromatic figures, repeated in a cyclical fashion, which globally move steadily downward in range, and finally die away. I realized much later after the recording that the results of her improvisation expressed the sentiment that I wanted during the final portion of the piece. These recordings were later transcribed and optimized for their use in the score.

The process of transcribing these chosen fragments of recorded improvisations into notation is similar to my use of various transduction methods to transform sounds. Transcription transforms the sound into a different and abstract medium, musical notation. This notation is again transformed when played back by the performer. The clarinet material played during the recording session has been transformed by the transcription into the score. Transcriptions of improvisations as fixed notation give the composer the ability to re-create the difficult to pre-compose, unpredictable, improvisatory spirit often contained in an inspiring improvised performance.
Several preliminary live performances of *Fractures* as an unfinished work were made with Dr. Yoder. These performances, each distinctive due to the inherent improvisational elements, also informed how I proceeded to shape the work over time. These preliminary performances and any future performances can be considered fractures that are a broken part of a larger whole that, for me, is the work.

Improvished performances using early versions of the computer software also inspired the creation of the work. The specialized software used in *Fractures*, in its generalized state, can also be utilized for solo laptop improvisation or laptop improvisation with chamber groups. In designing this computer program, my ultimate goal was to create a basic, flexible compositional environment for computer-based sound composition and improvisation. While developing the software for *Fractures*, in addition to performances with Rachel, I improvised on the computer with local improvisational groups such as Zanzibar Snails, and the University of North Texas-based Impulse ensemble. In the future, I intend to keep developing this software for my own future improvisational performance work.
Chapter 7

Software Development And Sound Design

*Physical Modeling with Waveguide Synthesis*

*Fractures* utilizes the physical modeling technique of waveguides synthesis to create clarinet-like timbres. Using Csound\(^\text{40}\) I created physical models based upon Perry Cook’s Synthesis ToolKit (STK)\(^\text{41}\) waveguide clarinet model that are controllable with a MIDI keyboard and continuous controllers. With the waveguide instrument, the clarinet’s timbral makeup can be explored by controlling individual features of the waveguide clarinet’s tone such as noise, vibrato, and reed stiffness. The waveguide instrument can be played manually using a keyboard controller with the output signal being recorded.

The following Csound orchestra and score were employed:

```xml
<CsoundSynthesizer>
<CsOptions>
  -Mn
</CsOptions>
<CsInstruments>
  nchnls = 1
  instr 1
    ; input
    iamp ampmidi 0dbfs; amplitude (scaled from 0 – 0dbfs, which is set to 32767 by default)
    kfreq init 0
    kmp linenr iamp, 0.5, -0.5, 0.01; envelope
    kstiff ctrl7 1, 91, -75, -0.05; negative values from -0.44--0.18 being the best
    knoise ctrl7 1, 93, 0, 1; noise gain from 0.5
    kvib ctrl7 1, 73, 0, 22
    kvamp ctrl7 1, 72, 0, 1
    kmod ctrl7 1, 1, 0, 127; modulation control
    iq = 50; filter Q
    ifmin = 400; min filter freq
    ifmax = 4000; max filter freq
    ival notnum
    kfreq cpsmidib ival
```


kmod = ifmin + (ifmax - ifmin) * (powtwo(kmod/127) - 1); scale MIDI modulation values (between ifmin and ifmax); using a logarithmic (equal-interval) scale
outvalue "amp", kmp
outvalue "cps", kfreq
outvalue "noise", knoise
outvalue "stiff", kstiff
outvalue "vib", kvib
outvalue "vamp", kvamp

a1 wqclar kmp, kfreq, kstiff, p4, p5, knoise, kvib, kvamp, 1
a2 reson a1,kmod,kmod/iq, 1; filter

out (a1*.95)+(a2*.05)
endin
</CsInstruments>
<CsScore>
1 0 8192 10 1
i1 1 999 2 2
 e
</CsScore>
</CsoundSynthesizer>

The waveguide clarinet offers an electronically synthesized clarinet tone with much greater timbral control than a simpler square-wave generator provides. Portions of the piece (mm. 68-79) utilize transformations between “abstract” or “pure” square wave sounds and more timbrally rich forms of synthesis using physical models to create diverse electronic metaphors for the clarinet.

*Max/MSP Real-time Sound Processing Algorithms*

Just as there is a diverse variety of sounds utilized in *Fractures*, there are also many different Max/MSP\(^\text{42}\) patches, external\(s\) and abstractions that form the real-time sound processing algorithms utilized in the piece. The work represents a culmination of various programs written in patching environments like Max/MSP and Pure Data (Pd)\(^\text{43}\) that I have utilized in several other compositions and studies since 1999. The programming employed in the work is primarily original with some adaptations of other publicly distributed

\[\text{42} \text{“Cycling 74 – Tools for New Media,” } \text{http://cycling74.com/} \text{ (accessed January 26, 2012)}\]
\[\text{43} \text{“Pure Data- PD Community Site,” } \text{http://puredata.info/} \text{ (accessed January 26, 2012)}\]
Max/MSP and Pd patches. Fractures' main program integrates these varied subpatches and
effects processing algorithms to work in tandem with a generalized and easily
programmable framework for real-time sound processing.

There are several components of this software that include relatively simple sound
processing. These include soundfile playback, delays, chorus, flanger, phaser, and
amplitude modulation. However simple, these sonic tools can be very powerful, especially
with the flexible control and signal routing paradigm that the Fractures software provides.
Fractures uses a signal routing paradigm that is directly based upon Cort Lippe’s crossbar
or “x-bar” system for controlling signal flow through Max’s matrix~ object.44 This system
provides very flexible control of signal flow that allows any source signal to be sent to any
signal processing algorithm. These controls can be used to create a flexible patch bay to
generate incredibly complex chains of signal processing. In addition, I would like to focus
on the more esoteric and unique sorts of sound processing programs that I have designed
for the work found in the real-time polyphonic recorder/sampler and harmonization
algorithms.

Real-time Polyphonic Recorder/Sampler

Recording live input(s) into any kind of buffer is a very useful and creative way to
store musical data for interactive music and live improvisation. This is especially helpful
when working with improvising musicians, who are often communicating with their own
personal musical language that could never be precomposed.

---

Signal Processing Workstation,” in Proceedings of the 10th Italian Colloquium on Computer Music, Milan, 1993,
430.
The recorder/sampler algorithm in *Fractures* consists of three main parts: buffer building, buffer recording, and buffer playback. These parts are co-dependent. Buffers must be created to allow real-time recording needed for buffer manipulation and playback.

*Buffer Building*

The **buffer_build** algorithm was designed with the help of Andrew May. Real-time digital sampling in Max/MSP requires the creation of a uniquely named buffer with a given size for each recording. Each buffer also needs to be large enough to hold its corresponding sample. The **buffer_build** abstraction provides a generalized way for the user to quickly modify these buffer parameters to suit different compositional needs for different pieces or improvisations. **Buffer_build** accomplishes this flexibly by allowing the user to, upon loading the patch, generate any given number of buffer instances with a base name and size in milliseconds. *Fractures*, using **buffer_build**’s abstraction arguments builds a reserve of sixteen different buffers, called “Buffer1, Buffer2, Buffer3, etc.” that are each twenty seconds long and are created when the main patch is loaded. Example 7.1 is the **buffer_build** abstraction that contains three creation arguments that represent, in order, the number of instances, base name for the buffers, and buffer size (in milliseconds).
buffer creation abstraction:
arg1 = number of instances;
arg2 = base name;
arg3 = size

Example 7.1: **buffer_build** Abstraction

**Buffer_build** utilizes the abstraction **Buffer_in_poly** (Example 7.2) that is embedded into the poly~ object. By embedding this abstraction into the poly~ object the user can create as many individual instances of this abstraction as necessary without having to manually duplicate them each by hand. The **buffer_build** patch triggers a sequence of events that (in right to left order) sends a size that is then stored as a placeholder in all of the embedded **Buffer_in_poly** abstractions, sends a base name that is then stored as a placeholder in all of the embedded **Buffer_in_poly** abstractions, and finally generates each one of the buffers by using the uzi object’s index number as a counter that counts up to the number of instances provided (the first creation argument).
Example 7.2: The First Instance of \textbf{Buffer\_in\_poly} Embedded Within poly~

\textit{Buffer Recording and Playback}

The recording paradigm for \textit{Fractures} consists of three key components: establishing the proper buffer where recording will take place, recording the incoming audio signal, and measuring the duration of the sampled recording. A global view of these functions is shown in Example 7.3.
Example 7.3: Global View of Fractures Recording Program

The transport for the recorder communicates with the program `gd_event_gather` and also directly communicates with `record~` to start and stop recordings. The transport controls for the sampler include basic controls such as start and stop buttons for recording. However, other more esoteric types of control are employed. One such control is the back button, which, if pressed after hitting stop, allows the user to re-record into the buffer they just recorded. The forward button skips over the buffer that is currently assigned for recording to the next one. A number box is also included with the comment “Choose One” that allows the user to control exactly which buffer they want to record. Finally, the clear button when pressed clears all the recordings held in the buffers at once. The only controls that are normally utilized in a performance of Fractures are start, stop, and clear. The back,
forward, and choose one controls can be utilized to correct mistakes the technician may make during a performance of the piece.

The recording number and the on/off state of the recorder are monitored and sent to another program, **buf_play_safe** (Example 7.4).

---

**Example 7.4: buf_play_safe Program**

This program ensures that the sampler does not play back buffers that are currently in the process of being recorded. By storing the buffer’s recording state into the table, **play_safety**, the buffer’s record state is always checked to make sure it is safe to play before playing back the sample.

When programming a polyphonic real-time recorder, it is helpful to keep track of attributes corresponding to each sample for future recall and transformation. **Fractures’**
recorder stores a recording index and duration for each individual recording using the program **gd_event_gather** (Example 7.5).

Example 7.5: **gd_event_gather** Program
A start bang marks the creation of a new recording, advances the recording count, and assigns a unique index to each recording. After all sixteen recordings have been made the count begins again and new samples are recorded over the old. A timer is utilized to measure how long in milliseconds each recording lasts. By combining the recording indexes and their corresponding durations and then storing this combination into a collection **gd_events**, a unique durational tag is given to each of the samples. This durational tag is employed in the software so that messages sent to the sampler’s polyphonic voices are always based upon their given length. Example 7.6 shows several durational tags stored in the **gd_events** collection.

1, 0 3239 3239;
2, 0 2298 2298;
3, 0 3303 3303;
4, 0 1721 1721;
5, 0 1248 1248;
6, 0 940 940;
7, 0 7598 7598;
8, 0 6878 6878;

Example 7.6: Duration Data Stored in the Collection **gd_events**

The durational tag is formatted for each sample as ([recording index #], [start time] [length in milliseconds] [ramp time in milliseconds]). This format facilitates direct use of the data to generate a ramp for line~ to play back the recorded sample at the original speed. The durational tag for each of the samples is integral to their method of sound generation. The sampler voices each use the object play~ driven by the line~ output. Playback speed, or sample transposition, is determined by the amount of time over which the samples are played. In order to play back a sample with a faster or slower speed one
alters the ramp time of line~ so that it does not equal the given sample’s duration. Faster ramp times will generate faster playback of the samples and vice versa.

The custom-designed sampler employed in *Fractures* is very flexible. The sampler has sixteen-voice polyphony with the ability to dynamically assign each of the recordings to any of its voices. For example, the user could play all sixteen samples back at the same time or they could have all sixteen voices assigned to play the same sample. The variable playback parameters include speed, direction (forward/ backward) volume, and fade length. There are also options to loop the playback of a sample or to play it back with a stereo panoramic trajectory, with a trajectory time corresponding to the samples’ duration.

The sampler produces audio when it receives a generalized formatted list that determines which voice is instantiated, which buffer will be played, and how the buffer will be played back. This generalized list can be sent by sending a new voice number utilizing the sampler control GUI (Example 7.7) or a by sending the sampler an appropriately formatted list.

![Sampler Control UI](image)

**Example 7.7: Graphic User Interface for Sampler Control**
When using this GUI for controlling the sampler, various chosen parameters are sent to the sampler whenever a new voice number is selected. The lower right portion of the GUI contains controls that locally modify the sampler control parameters. Currently, there are two modifiers called **metroplex** and **random transposer**. The **metroplex** is a metronome-based sample playback algorithm that allows the user to control the metronome speed and the pattern of samples that are played back. The **metroplex**'s range is modified so that it expands accordingly for each new sample that is recorded. In the number box labeled “random/ drunk/ pattern” there are three different options: if 1 is selected the samples playback randomly, while selecting 2 results in a drunk walk, and 3 generates pattern-based selection based upon the modifiable table **gd_patterns**. The **random transposer** adds randomness to the transposition parameters for sample playback. It has controls for range and transposition offset (base transposition). There is also a toggle control that forces the modifier to output integers, which constrains the random transpositions to equal temperament. When using the **metroplex** modifier each new sample is sent to a voice number by utilizing a counter that counts perpetually from one to the maximum number of voices.

In addition, lists can be sent to the sampler formatted as ([voice #] [buffer #] [transposition] [reverse toggle] [loop toggle] [fade in/out time] [pan start value] [pan end value] [pan time] [output volume]). The program **stop_gestures**, in addition to stopping the recorder, sends lists to the sampler whenever a stop gesture cue is received (Example 7.8).
Example 7.8: **stop gesture** Program for List-based Control of the Sampler
The **stop_gesture** program receives both the current recording number and voice count so that when a stop gesture is cued, the last recorded sample is played with the next available voice.

*Polyphonic Voice Allocation*

Voice allocation is another important factor to the design of a polyphonic sampler. *Fractures* utilizes the program **gd_playback_controller** to assist with routing messages to each of the sampler voices (Example 7.9).

![Diagram showing polyphonic playback](image)

**Example 7.9: gd_playback_controller Program**

This program utilizes the sprintf object to create dynamic communication with specific sampler voices. Currently, Max/MSP’s send object’s register name cannot be dynamically altered, but by utilizing both sprintf and the forward object, a unique register name can be set dynamically for sending messages. Whenever a sample message is called, three things occur. First, the voice number from the formatted list is used to dynamically set the register for all the forward objects. For example, if voice number six is called, then all the sprintf objects add the number “6” at the end of their string and dynamically set the registers for their corresponding forward objects. Second, all the parameters from the list
are sent through the appropriate registers to the correct sampler voice. Third, the buffer number followed by the voice number are sent to the algorithm `Play_poly` for sample playback (Example 7.10).

Example 7.10: Buffer and Voice Values Sent for Sample Playback

When the voice number is finally sent it sets the target number for which voice will be played and then bangs the chosen buffer number. This buffer number is then sent into the collection `gd_events`, which has been filled with durational tags for the corresponding buffer by the `event_gather` algorithm.

There are sixteen copies of the `Play_poly` algorithm (Example 7.11) embedded in `poly~` that `gd_playback_controller` communicates with.
Example 7.11: **Play.poly** Algorithm for Sampler Voices

**Play.poly** uses the thispoly~ object to monitor which particular poly~ voice it is utilizing and sets its receive registers accordingly with a string generated by sprintf. This creates a unique set of receive registers for the particular voice that correspond to parameters sent by **gd.playback_controller**. Parameters are sent to the appropriate voice and affect the method of sound generation locally within **Play.poly**. There are algorithms within **Play_Poly** that assist with the sound generation. These include **event_mangler** (Example 7.12), which modifies the list data received from the **gd.events** collection so that changes in transposition or reversal are made.
Example 7.12: event_mangler Algorithm for Modifying List Data

In addition, algorithms for creating a playback envelope using curve~ and panning are also included locally within the Play_Poly sampler voice.

Programming a real-time polyphonic recorder and sampler with a great amount of flexibility and control can be quite complex. However, many of the unique problems such as buffer creation, sample recording, sampler control, and voice allocation have each been addressed in this study.
Harmonization Algorithms

Fractures makes use of several unique approaches to real-time harmonization of the clarinetist, through the use of real-time analysis tools, a pitch-sieve algorithm written by Jody Nagel, and a low-frequency oscillator controller. Fractures’ harmonizer is also used for fixed transposition harmonization as well.

The smart_harmonizer algorithm (Example 7.13) is a real-time effect that combines a real-time pitch analysis tool, analyzer~\(^{\text{45}}\), by Tristan Jehan, Nagel’s PitchSieve, and a fast-Fourier transform (FFT)-based harmonizer, gizmo~, by Richard Dudas.\(^{\text{46}}\)

---

Example 7.13: The **smart_harmonizer** Algorithm


The **smart_harmonizer** analyzes the incoming pitch of the clarinetist using analyzer~. This analysis data is then filtered to exclude erroneous pitch data below the clarinetist's range and rounding the pitch to the appropriate semitone value. The pitch can then be transposed before reaching the pitch sieve. When the **PitchSieve** algorithm is given a pitch-class list one to twelve pitches long, it constrains the incoming notes to match its pitch-class list. **PitchSieve** parsimoniously shifts the incoming notes that are outside its given pitch-class list to the closest relative pitch class that is in the list. If two notes in the pitch-class list are equidistant from a “wrong” incoming note, **PitchSieve** will choose randomly between the two. After **PitchSieve** outputs its selected pitch the original pitch is subtracted from the output to calculate the difference between the two pitches. This pitch difference is the positive or negative transposition applied to the harmonizer for harmonization corresponding to **PitchSieve**'s output.

Richard Dudas has explored a similar type of harmonization effect in his work *Prelude for Clarinet and Computer* (2006).\(^47\) Dudas writes:

> The most important development for the *Prelude for Clarinet and Computer* takes a small inspiration from the world of algorithmic composition, albeit performs it in the context of a digital signal processing effect. The author designed a pitch-class corrector as an event-level Max patch, whereby all incoming (MIDI) pitches are modified to fit to a pre-defined pitch grid. Each input note is compared to a given pitch-class set, and then transposed to the nearest note in the set. When used in conjunction with the robust ptg~ pitch tracker and the gizmo~ object, each note

---

played by the clarinet may be adjusted to a different fundamental frequency in real-
time. In some ways this is the antithesis of the ubiquitous auto-tune effect in
popular music, as it does not correct the fine tuning of the note, but rather moves
the entire note event to a different pitch-class level, preserving vibrato and
microtonal nuances inherent in the original sound.\footnote{Ibid., 71-2.}

The preservation of timbre of the original source sound in the processed output of a
harmonizer and the other real-time effects utilized in \textit{Fractures} is a very attractive benefit.

In addition, the data output of a low-frequency oscillator (LFO) can be added to
create another layer of transposition control after the processing with

\texttt{gd\_smart\_harmonizer}. The program \texttt{event\_sine\_osc}, written by Andrew May, is utilized
within the patcher \texttt{Ifo} to create two different event-rate LFOs that output typical MIDI
values (0-127) (Example 7.14). One of the LFOs functions as a carrier oscillator and the
other a modulator. These oscillators are both typically set with very low frequencies and
ranges to create subtle sweeping changes of pitch over time.
Example 7.14: The **Ifo** Patcher

The carrier **event_sine_osc** oscillator's output is routed into a separate pitch sieve. The output of this pitch sieve is transformed just as in **gd_smart_harmonizer** by subtracting the LFO’s original pitch from the pitch sieve’s output to find the positive or negative transposition applied to the harmonizer corresponding to **PitchSieve**’s output.
The carrier oscillator's period determines when changes in the pitch-class structure of the harmonizer occur. This period is calculated in the lfo program (period in milliseconds = \(1/\text{frequency} \times 1000\)). The phase output of the carrier LFO can be utilized for other applications as well. For example, the send labeled lfo_sig is utilized during mm. 32-44 to create oscillating interpolations between two different effects matrix presets.

*Structural Design of the Main Performance Program*

*Fractures' main performance program* (or patch) is designed to be a flexible, generalized, software framework that can be employed for compositions and also as a live improvisation tool.

Controlling a wide array of variables over time in a flexible manner is perhaps the most fundamental goal in creating complex interactive electronic music using object-oriented programming environments. Some parts of the program (the low-frequency oscillator, real-time effects processing algorithms, DSP matrix-controls, and polyphonic recorder/sampler) each have the exceptional ability to be controlled by a flexible preset system. This preset system is created around Max's pattr paradigm for controlling many different variables at once. A separate pattr preset system is utilized for each portion of the program.

Programming patchers that utilize pattr requires the programmer to give a unique scripting name to each variable manually, providing greater flexibility and ease of use. Traditional methods for controlling presets in Max include using the preset object or utilizing a qlist or message boxes to send cues to multiple registers of receive objects. In addition to the benefits afforded by those methods, pattr-based preset control has the added ability to flexibly interpolate between values of two different presets. The pattr-
based preset system also stores this control data in a separate file that is saved onto the computer. This file can then be edited using a text editor if desired.

The pattr program for DSP matrix-controls features almost one hundred different variables (effect sends) that are controlled by this preset. Each of these variables functions like an effects send-knob on an analog mixing board. These effect sends control the gain of signal pathways utilized in Fractures X-bar-based signal routing paradigm (discussed in Chapter 7). By interpolating between different routing settings a crossfade between multiple effect send variables is possible.

Pattr provides the same ease of control as Max’s preset object, but without many of the problems associated with it (lack of specificity, flexibility, control, etc.). This encourages electronic composers to listen and tweak effects parameters, finding interesting parameter combinations and immediately storing them as presets.

Using qlist or message boxes to send wireless messages to various registers of receive objects gives specificity, flexibility, and control of presets. However, this method lacks the spontaneity offered by the pattr system in that it requires that every single send/receive register that is changed by a preset be accounted for in the qlist manually. If Fractures utilized this approach to writing a qlist, there would have to be hundreds of lines written with send register names and values for each of the changing parameters for every single cue. Rather than performing this painstaking task for each cue, Fractures qlist can control hundreds of parameters at once with just a few short written commands that change pattr presets. Example 7.15 shows a variety of qlist commands for cues 11-15.
0 11 ----------------------- 11;
fx_matrix_preset 1 10000;
sf_num 3;
0 12 ----------------------- 12;
sampler_preset 2 0;
sf_num 4;
0 13 ----------------------- 13;
sampler_preset 3 18000;
sf_num 5;
0 14 ----------------------- 14;
sampler_preset 1 0;
fx_matrix_preset 2 5000;
0 15 ----------------------- 15;
fx_matrix_preset 4 5000;

Example 7.15: Qlist Commands for Cues 11-15.

This qlist accomplishes several different tasks. For example, cue 11 shifts the DSP matrix-controls’ pattr preset by sending a new cue to the register named \texttt{fx\_matrix\_preset}. The numbers following \texttt{fx\_matrix\_preset} represent the cue number, and the amount of interpolation time (in milliseconds) taken to reach the new preset state. The command \texttt{sf\_num 3} in cue 11 starts the playback of the third soundfile from the soundfile player. The \texttt{sampler\_preset} register controls an entirely different pattr system that controls the sampler. This register controls the preset with the same generalized controls as \texttt{fx\_matrix\_preset}: a preset number followed by a preset interpolation time (in milliseconds.) All four of the pattr preset systems utilized in the work employ this same generalized message system.

Historically, for interactive works using Max/MSP the qlist represents an electronic “score,” in that the qlist generally charts the various changes that occur in the computer’s electronic part. In \textit{Fractures}, because of the use of several pattr preset systems, this particular qlist presents a much more skeletal view of what is actually occurring in the
electronics. If one wanted to actually analyze the many local changes that occur within each particular preset, they would have to look at the details of the individual stored pattr presets.

This birds-eye view of the qlist is advantageous because it immediately shows a bare representation of what is going on compositionally, while a lot of compositional minutiae can still be analyzed in more detail by consulting the pattr presets. This very quick approach to storing presets and recalling presets also encourages improvisation and further composition with the system. Composers are often limited by the speed at which they are able to manipulate tools that aid their musical imagination. Composing complex transformations of sound and sound processing utilizing Fractures generalized qlist/ pattr system is intuitive and fast. This compositional approach attempts to mimic the intuitive and corporeal nature of music created by early electronic musicians.

Graphical User Interface Design of the Main Performance Program

The GUI design of the main performance program includes all the necessary controls for the technician to perform the work and receive feedback from the program to ensure a successful performance. There are also several features that enable the technician to rehearse the piece non-linearly with or without the performer and record their performance. Example 7.16 shows the main performance program's GUI.
Example 7.16: Graphical User Interface of the Main Performance Program

The GUI displays each of the controls with its own unique color for easy identification. Each control feature is embedded within a bpatcher object, which allows the inner complexities of the controls to be hidden. These GUI controls often interact directly with the programs embedded in patcher objects. For example, the sampler transport and control are strictly controls for the patcher sampler, while the cueing interface controls the qlist object contained within the patcher, qlist. The sampler control GUI also allows the technician to watch samples being recorded in real-time using the waveform~ window. This can be very useful to monitor sound levels over time and also to make sure the length of the samples being recorded fits into the available buffers.

The input/output section on the far right is used to control levels for input and output signals as well as monitor these levels. The input fader labeled “mock clarinet” is used by the technician for monitoring a pre-recorded clarinet part during rehearsal and
testing without a performer. In addition, there is a toggle control that changes the analog output of the entire system from stereo to four channels. When this switch is made, the sampler and soundfile player’s outputs are sent to soundcard outputs three and four. This option gives the technician utilizing an analog mixer the ability to adjust the balance between the stereo playback of the real-time effects algorithms with the stereo playback of the sampler/ soundfile player during performance. The performance of the electronics can be recorded in both two-channel and four-channel modes.

Most of the programs utilized by the main patch’s GUI are embedded in several different Max patches. While their inner complexities remain hidden, the patchers’ contents can all be accessed easily via the main performance patch. Together these programs comprise the digital signal processing (DSP) setup, DSP matrix, DSP matrix-controls, polyphonic recorder/ sampler, real-time effects processing algorithms, soundfile player, low-frequency oscillator, qlist-based cueing system, and tools supporting both MIDI and QWERTY keyboard control of the system. These various programs each have a clearly defined function and are given patcher names that clearly describe what they accomplish. Each of these programs can easily be accessed via the main performance patch.

*Fractures* contains the patcher, **rehearsal**, for technical rehearsals of the work when a performer is not available (Example 7.17).
Example 7.17: **rehearsal** Patcher

This program sends a list of numbers to the patcher, **mock_clarinet** (Example 7.18).

Example 7.18: **mock_clarinet** patcher
The **mock clarinet** patcher utilizes an sfplay~ object and its associated sflist~. A pre-recorded clarinet file, “Faux_Fractures_RNP+30dB,” is included with the software and is preloaded with a set of defined cues using sflist~. These cues define where the recording will start playing back when a list is sent to **mock clarinet**.

The first number of each list is also sent to the collection, **mock cues**, which recalls the correct cue associated with the index number.

The cueing interface in the main performance patch allows the technician to start from any cue and also to monitor which cue they are currently on. Due to the nature of the work there are a number of performance problems that can occur when rehearsing *Fractures* in a non-linear manner. Both sections B and C require the technician to sample live recordings of the clarinetist. If the technician were to begin rehearsal right on cue 17, which occurs after several samples are recorded and starts playback of the sampler, the sampler won’t have any recorded samples to play. For this reason, rehearsable cues in the score are placed within a pentagon shape and non-rehearsable cues are placed within a square.

The GUI of the main performance program provides an optimal environment for performance and rehearsal of *Fractures*, with or without a performer. The program provides easy access to all of the necessary controls that the technician needs and provides the feedback required to ensure a successful performance of the work. The elements of the work that do not need to be seen during performance are embedded within patchers. These patchers each have clearly defined names and function and can be easily accessed directly from the main patch.
Conclusion

*Fractures* brings together many diverse musical materials and electronic music techniques. This includes the transformation of borrowed samples and scores and the utilization of varied acoustic spaces, hardware, transducers, and computer sound synthesis and processing techniques. The exchanges between these diverse elements and the performer are explored through the interactive musical environment designed to flexibly interact as both an accompanist and equal partner. *Fractures’* rhapsodic form is created by the dramatic shifts between sections that result from contrasting musical traits. *Fractures’* real-time sound effects processes and distinctive approaches to real-time sampling and harmonization are dynamically shaped throughout the work, creating a dialogue with the performer. A generalized software framework controls these effects, making highly complex composition or improvisation with the software both intuitive and fast. In addition, *Fractures’* software provides an attractive graphical user interface that aids its use for future composition, improvisation, and performance.

*Fractures’* generalized and robust global design provides flexibility and ease for implementing new developments. I would like to continue to use this global framework for future works and improvisations. However, there are several parts of the program that I would like to see expanded in the future. These developments include additional physical controls to manipulate the software instrument, real-time effects, algorithms to alter sampler playback, global controls for signal routing and effects architecture, and the integration of interactive physical models and concatenative synthesis.

Physical controls for the software could be expanded. Sampler playback utilizing a MIDI controller would be very useful for improvisations. In addition, greater control of
real-time effects and signal routing would offer many exciting new possibilities for interaction with the software. Both real-time effects and signal routing supply many different parameters. Consequently, the system would need to be designed to allow each of the controls to be extremely flexible. The program could be built to include customizable MIDI or OSC\textsuperscript{49} control mapping for any variable(s). The user could be given intuitive control of many parameters at a time using a single multi-dimensional control structure.

Additional real-time effects are simple to integrate into Fractures’ flexible signal routing scheme. I would like to continue to add real-time effects such as convolution, time expansion/compression, granular synthesis, distortion, spectral filtering, and spectral delay that would each be able to interact with the currently available effects.

There are several future developments that could be made to Fractures’ polyphonic recorder and sampler. Many new algorithms and stop gestures could be included to manipulate the sampler’s playback behavior. Additional analysis of recordings beyond just their duration could create many possibilities for interactive control of sample playback. Properties of samples such as their initial pitch, average pitch, initial amplitude, average amplitude, peak amplitude, and noisiness could all be recorded along with each of the samples into tables or arrays. These analyses might also provide helpful, intuitive feedback for the performer improvising with the software.

Another improvement that I would like to see implemented is global control of the program’s signal routing and effects architecture. Currently, while it is quite simple to add a new effect, sound-generating instrument, input channel, or output channel into Fractures signal routing scheme, from a programming standpoint it is very time-consuming. Due to

the nature of the crossbar control of matrix~ employed, when a new effect or voice is added, signal routing possibilities expand exponentially. This also means that the time it takes to program all the new signal pathways and pattr scripting names expands exponentially. What would be more ideal is a way of providing a set of generalized instructions to Max/MSP about the user’s desired general architecture and have Max/MSP generate that desired architecture utilizing live coding. For example, the user could request a number of analog input channels, number and types of sound generators (like the polyphonic sampler or soundfile player), number and types of real-time effects, and number of output channels. Max/MSP would then generate all of the programming necessary to create the proper signal routing matrix, provide a manual slider control for every signal pathway, and supply the scripting names necessary for pattr preset control of the entire system. This would allow the system to be quickly customizable and efficiently streamlined to meet different compositional and performance needs.

Interactive physical models and concatenative synthesis would also be attractive sound generators to add to the software system. During the composition of Fractures, I worked towards integrating an interactive waveguide clarinet model into the Max/MSP environment using the csound~ object. My development shows promise, but additional research and development is needed to improve the instrument’s subtlety and nuance. Concatenative synthesis proved to be a very attractive sound processing method for the work. Real-time interactive control of concatenative synthesis may be a promising new area to explore further in future works.

---

Fractures’ sampling and borrowing of other composer’s works proved to be an exciting compositional choice. This material enables a composer to create associations with historical culture and its associated collective memory. However, it can be challenging to create a new piece of music with strong references to other works in a respectful manner. Sampling of popular music can be a powerful tool to create associations for a listener, but can also potentially distract them from the newer work. Fractures imports the vintage sonic character of early popular electronic music while addressing these problems by fragmenting and continuously transforming the original materials.

I believe my compositional work will continue to borrow ideas from other composer’s music as models or archetypes. I intend to continue to write electronic works focusing upon surreal transformations of sound, space, and fidelity. I would also like to continue to write works that focus upon older forms of music technology. I hope the clarity of the score and ease of use of the software will lead to many more performances by other clarinetists and sound technicians. Fractures represents a culmination of my composition and computer music research over many years. Additionally, the piece also represents a new foundation for my future interactive improvisations and compositions that will employ a similar global framework and software design.
Selected Bibliography


Carlos, Wendy. Switched on Bach, Columbia Records MS 7194, 1968. LP.


Lazier, Ari and Perry Cook. “Mosievius: Feature Driven Interactive Audio Mosaicing.” In


Moog, Robert. Moog. Directed by Hans Fjellestad. 72 min. ZU33, 2004. DVD.

O’Rourke, Jim. Please Note Our Failure. Some some-08, 1998. 10” vinyl.


Oswald, John. Plunderphonics. Plunderphonics Mystery Tape Laboratory WRC1-5744, 1988. LP.


Scott, Raymond. *Soothing Sounds for Baby Volume 1: 1 to 6 Months.* Epic LN 24083, 1962. LP.


PART II

FRACTURES FOR CLARINET AND COMPUTER
Greg Dixon

Fractures

for clarinet and computer (2012)
Program Notes

Fractures celebrates nostalgia and the past, while recontextualizing the past into something completely new. The work appropriates ideas from acoustic works for clarinet by composers such as Gershwin, Tchaikovsky, and William O. Smith. At times, the work is a rhapsodic fantasy inspired by the many popular synthesizer albums from the 60's and 70's created in the wake of Wendy Carlos' seminal album Switched on Bach. These sections contain very brief sampled quotations from LP's of electronic works that explore the synthesis of clarinet-like tones. These quotations are borrowed from popular recording artists of the time, such as Wendy Carlos, Dick Hyman, Ruth White, and Mort Garson, along with many others. Samples of vintage synthesizers are also a part of this sonic palette. Mannerisms and styles from these works also influence the score for the clarinetist.

Performance Notes

Notation
Accidentals are maintained throughout each bar and are canceled by barlines.

If no meter is given there are two possible options for tempo interpretation:

1. A bracket provides the total duration of the bar or system. Bracket subsets may also be specified. Stems or flags provide relative duration.

2. Rhythms are played based upon the specified tempo without any downbeat accentuation.

Technical Notes

The computer part for Fractures was written using the software Max/MSP (version 5.18). This version (or a later version) of Max/MSP or Max/MSP Runtime should be used to perform the computer part.

The piece requires one microphone to amplify and process the sound of the clarinet. Quality condenser lavaliere microphones are preferable.

Fractures utilizes a variety of computer cues. There are cues for the following: Events, "Start Recording," and "Stop Gestures." Stop Gesture cues accomplish two different things: 1. they stop the recording and 2. they play back the prerecorded sample in various prescribed ways.

The technician can use a MIDI controller or ASCII keyboard to control the Start Recording and Stop Gestures used in this piece. The Max/MSP patch for fractures contains a patcher 'setup_MIDI' where one can modify the MIDI messages that control the Start Recording cue and the various Stop Gesture cues. The patch can also set it up so a computer keyboard can be used to accomplish these cues. To turn the ASCII keyboard control on, turn on the toggle switch located above the patcher, "keyboard."

Stop Gesture Key (Stop Gestures a-h are keys 1-8 respectively)

Stop Gesture a (1): immediately plays the prerecorded sample at normal speed.
Stop Gesture b (2): immediately plays the prerecorded sample transposed up an octave.
Stop Gesture c (3): immediately plays the prerecorded sample transposed down an octave.
Stop Gesture d (4): immediately plays the prerecorded sample transposed randomly within a two-octave span.
Stop Gesture e (5): immediately plays the prerecorded sample backwards at normal speed.
Stop Gesture f (6): immediately plays the prerecorded sample backwards transposed up an octave.
Stop Gesture g (7): immediately plays the prerecorded sample backwards transposed down an octave.
Stop Gesture h (8): immediately plays the prerecorded sample backwards transposed randomly within a two-octave span.

Notation of Cues for Computer

Events:
Clear Buffers:
Rehearsal Cues:
Non-rehearsal Cues:
Start Recording:
Stop Gestures:

Clear (delete)
Stop (backslash)
Fractures
for Rachel Yoder

Greg Dixon

\[ \text{Clarinet in B}^b \]

\[ \text{Computer} \]

\[ \text{CPU} \]

\[ \text{Cl.} \]

\[ \text{CPU} \]

\[ \text{Cl.} \]

\[ \text{CPU} \]

\[ \text{Cl.} \]

\[ \text{CPU} \]

\[ \text{Cl.} \]

\[ \text{CPU} \]
Fractures

Cl. CPU
Similar texture continues
phasen off
harmonizer/delay on (- tritone)

c. 10"

Mp mf mf mf f mf pf

Texture begins to dissipate
dim.

Harmonizer/ delay off
Phaser on
c. 10"

Harmonizer/ delay
on (- tritone)
Phaser off
c. 10"

Harmonizer/ delay
(*) m3

78
Choose any of the following five systems, play it completely, then move on to another. Continue after playing all systems.

Choose any of the following five systems, play it completely, then move on to another. Continue after playing all systems.
Fractures

Cl. 28

CPU

speeds up
algorithmic
playback of
sampler

Cl. 29

CPU

Cl. 30

CPU

Free/ Floating

Cl. 31

CPU

gradually stops
playback of
sampler
phaser on

Cl. 32

CPU

amplitude
modulation on

Cl. 33

CPU

Cl. 35

CPU

Stop

Stop

Stop
Fractures

Cl. CPU

36

\[ \text{mf} \rightarrow \text{mp} \rightarrow \text{f} \]

37

\[ \text{p} \rightarrow \text{mp} \rightarrow \text{mf} \rightarrow \text{p} \]

38

\[ \text{f} \rightarrow \text{p} \rightarrow \text{mf} \rightarrow \text{p} \]

39

\[ \text{mf} \rightarrow \text{p} \]

40

\[ \text{mf} \rightarrow \text{p} \]

41

\[ \text{f} \rightarrow \text{mp} \rightarrow \text{f} \rightarrow \text{p} \]

42

\[ \text{pp < mf} \rightarrow \text{mp} \rightarrow \text{mf} \rightarrow \text{p} \]

43

\[ \text{f} \rightarrow \text{mp} \rightarrow \text{f} \rightarrow \text{p} \]

Cl. CPU

17

starts algorithmic playback of sampler

18

soundfile #3

19

speeds up gradually increases transposition of sampler harmonicizer + TT (lfo off)
Fractures

Cl.

CPU

Exorcism

Cl.

CPU

soundfile #4

Cl.

CPU

soundfile #6

Cl.

CPU

like a siren

texture begins

Cl.

CPU

harmonizer/ delay

(+ m3)
soundfile #6

hi on
modulating
harmonizer
Fractures

rit. poco a poco

Fractures

reset LFO
for the next cue

expressivo

Free/ Floating

harmonizer/ delay

LFO on (- tritone)

modulating

harmonizer

LFO's amplitude
gradually decreases
over then next 2 minutes

(continues &
slowly dissipates)

(fixed media
stops)

85
Fractures

soundfile #7
'sparkly'
synth texture enters

phonograph
needle across
old and empty
grooves

needle reaches the end of the LP