NORTHWESTERN UNIVERSITY

A Simplified Interface for Granular Processing Based on Perceptual Research

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Music Technology

By

Nathan Edwin Wolek

EVANSTON, ILLINOIS

December 2005

ABSTRACT

A Simplified Interface for Granular Processing Based on Perceptual Research

Nathan Edwin Wolek

Granular processing is a computer music technique that manipulates "grains" of sound to produce a variety of effects. Grains are produced by multiplying short segments of digital audio, typically lasting between 10 and 50 milliseconds, with an amplitude envelope of equivalent length. Software designed to produce granular processing effects often requires the user to manage multiple parameters that lack a clear connection to the audio output. A better understanding of how listeners perceive the processing output should yield insights into how the user interface could be simplified.

A series of three experiments was designed to investigate how listeners perceive differences between granular processing examples. Stimuli were produced using specific program settings to process two distinct sound sources. In each experiment, at least twenty participants were asked to rate the similarity of each possible pair of stimuli including identity pairs that existed among these stimuli. The author then used multidimensional scaling (MDS) to develop a graphical representation of the perceptual organization exhibited by participants.

Differences between stimuli included the processed sound source and settings for the grain duration and grain period parameters. Visual analysis of the MDS solution showed that participants clearly distinguished between the two sound sources. Processing descriptors based primarily on the review of literature were tested for correlation to the MDS dimensions. This analysis revealed the significance of three processing features: (1) a base-2 logarithmic scaling for differences in grain duration, (2) the minimum and maximum boundaries for randomized grain durations, and (3) the mean value and total deviation for randomized grain periods.

Between-subject variables relating to experience with electroacoustic music were also examined. However, the results of this secondary inquiry were deemed inconclusive overall based on the relationship between participants' responses to pre-experiment questions and a priori operational definitions.

The findings were used to inform the design of a new graphical user interface (GUI) for granular processing. The resulting GUI helped to verify this study's conclusions by successfully demonstrating their practical application to software development. The GUI features unique controls for managing randomization and a feedback display for monitoring differences between the control input and audio output.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Granular processing (Roads, 1985; Truax, 1987) is a computer music technique that has experienced increased interest in recent years. It is part of a group of techniques united by their use of short bursts of sound called "grains" or "particles", a concept traced back to Dennis Gabor's premise of *acoustical quanta* (Gabor, 1947). Curtis Roads chronicled the collective development of these granular techniques in his recent book *Microsound* (2001) and asserted that they "now stand at the forefront of compositional interest" (p. 21). In response to this interest, a growing amount of software has become available to create granular processing effects on contemporary computing platforms (e.g., Bencina, 1998, 2001; Nelson, 2000a; Rolfe and Keller, 2000; Tanghe, 2003; Wolek, 2001).

The increased interest in granular processing has occurred despite the steep learning curve often associated with the technique. Roads (2001) noted, "it takes a certain amount of training to learn how operations in the micro domain [where grains occur] translate to acoustic perceptions on higher levels" (p. 26). Granular processing uses massed controls to manage the low-level details for hundreds of grains simultaneously. Truax (1988) explained that each of the "control variables" has "a psychoacoustic correlate that may be more suggestive as a basis for compositional organization" (p. 18). These statements by Roads and Truax both acknowledged a complex relationship between the user's control input and the resulting acoustic output. The user must learn how control input "translates" into the resulting sonic output, creating a significant source of confusion for new users.

A less ambiguous relationship between the control input and audio output could reduce the confusion currently experienced by many users. A necessary precursor to developing such a relationship is a clear understanding of how the audio output is perceived. In research to date, conclusions about the perception of granular sounds have been made based on preexisting *psychoacoustic* research. For example, Gabor (1947) cited a 1935 study by Bürck, Kotowski and Lichte and a 1931 study by Shower and Biddulph in his attempt to identify a minimum duration for grains. Truax (1988) connected the perception of granular sounds with the concept of *streaming* (McAdams and Bregman, 1979). Xenakis (1963/1992) sought to make changes in his control parameters that would compensate for Fletcher-Munson equal-loudness curves. While such informed decisions are a positive first step, granular sounds have now reached a level of usage that warrants the attention of an empirical research project. To date, no direct study of the high-level acoustic percepts related to granular sounds has been conducted.

This document details the development of a new user interface for granular processing based on the findings of three perceptual experiments. This empirical research was designed to facilitate direct observation of the differences subjects were able to detect between unique examples of granular processing.

Wessel (1979) demonstrated the use of empirical research as the basis for computer music software design, successfully developing an interface for the musical control of timbre. Using multidimensional scaling or MDS (Kruskal, 1964a, 1964b; Shepard, 1962a, 1962b) to analyze similarity ratings provided by his participants, he developed a two-dimensional map that expressed the apparent relationships between 24 orchestral timbres. This map formed the basis for an interface control, through which the composer could navigate to create interpolated timbre changes between the discrete points representing the original timbres. The findings of his experiments allowed Wessel to develop a program interface that successfully met the composer's needs with great economy. It is Wessel's study and its unique strategy for software development that has inspired the design of the current study.

This document is intended for readers already familiar with computer music practices who may be interested in new approaches to designing audio processing software. The study focuses on a computer music technique known as granular processing, the specifics of which will be explained. This document will also be of interest to those who are familiar with music cognition research. Wessel's 1979 study is part of a larger body of research that has studied timbre perception through the use of MDS (e.g., Grey, 1977; Iverson & Krumhansl, 1993; Kendall & Carterette, 1991). Those familiar with these timbre research methods may be interested in how they are applied within the current study, however the results are not easily related to studies focusing on the timbre of orchestral instruments. The application of these empirical research methods to problems of computer music software development would be the primary point of interest for such readers.

The organization of this document should enable readers to clearly understand the context of the empirical study that the author has conducted. The first chapter describes how computers are used as part of the music composition process. Chapter 2 recounts the development of granular processing and defines related terminology. Chapter 3 contains a survey of software that has been used to produce granular sounds with analysis of each interface. The fourth chapter details the design and execution of the three perceptual experiments at the core of this study, with chapter 5 providing analysis of the results. Chapter 6 describes how the analysis has informed the author's new interface for granular processing and suggests future directions in which this research might proceed. Should individual readers deem portions of this study outside their interests, the structure should enable them to skip chapters as needed. Those comfortable with the basics of granular processing may wish to skip chapter two. Those uninterested in the details of the experimental procedure may want to skip chapter four and continue with the analysis in chapter five. This study is situated between two domains of research: computer music programming and music cognition research. The organization of this document should help readers from both fields focus on the elements of interest to them.

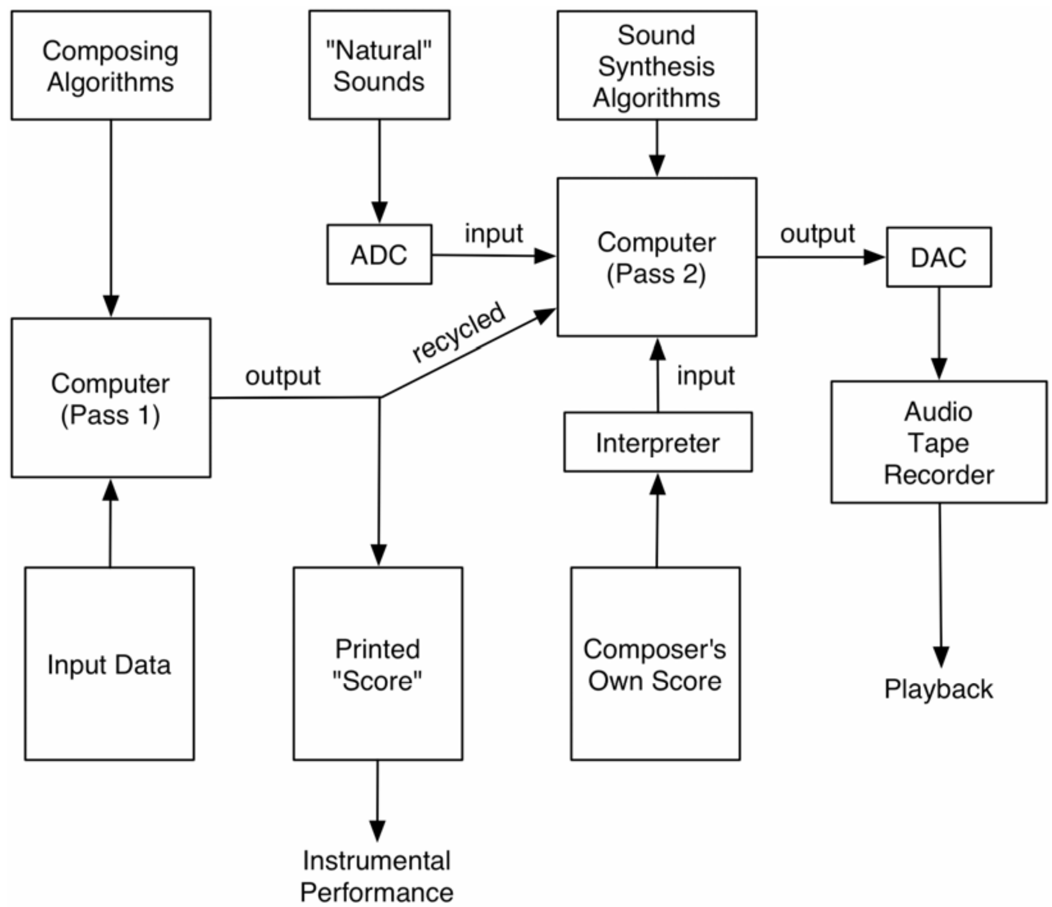
CHAPTER ONE: PROGRAMMING FOR COMPUTER MUSIC

A computer is incapable of performing any task without a properly defined set of instructions known as a *program*. Developing such instructions sets for a computer is referred to as *programming*. Chowning (1996) noted, "Programming involves mental processes and rigorous attention to detail not unlike those involved in composing" (p. ix). It is perhaps for this reason that certain composers have become adept at creating their art with computers. However, "computer music" is a term that has much broader implications today than it did in the past. Personal computers with commercially available programs are now the normal method by which musical recordings are captured and produced for distribution. Chadabe (1997) stated, "one could say that by the late 1980s the age of computer music was over because *everything* was computer music" (p. 139). Lyon (2002) later suggested, "the prevalence of the use of computers in today's music demands another distinction; at its outset computer music meant experimental music, carried out in laboratories and universities." Lyon saw this "experimental music" as distinct from what he called "normative music," which he defined as "music based on accepted stylistic norms" (p. 13). The distinction is an important one. It is experimental computer music composers who are typically also programmers, performing both activities interchangeably in order to realize their works.

A. Musical Tasks Performed by Computers

In order to understand the computer's role in this type of experimental music composition, it is useful to first identify the musical tasks that computers are typically programmed to perform. Hiller (1989) divided computer music into two types of tasks: "computer-composed music" and "computer-realized music." In the first type, composers use algorithms to determine how the elements of a musical score are organized. Hiller's 1957 composition *Illiac Suite for String Quartet* was an archetypal piece of computer-composed music. In order to produce the musical material for the score, Hiller programmed a mainframe computer to calculate variations in pitch, duration and timbre based on random number sequences. In the second type of task, composers use computers to synthesize and transform digital audio for the eventual transduction into sound. Computer-realized music is an extension of the techniques used for electroacoustic music. Truax (1986a) noted, "A fundamental trait of the practice of electroacoustic music is that the composer composes the sound itself as well as the structure in which it appears" (p. 156). Hiller (1989) also attempted to model how a composer might perform both tasks in the creation of a piece (see Figure 1). His model connected the two tasks in series, with each realized during a separate "pass" through the computer. Although he clarified the model, stating that "the boundary between these two processes can be sharply defined or rather fuzzy" (p. 75), the sequence of Hiller's tasks showed the influence of traditional

Figure 1. The computer used as a composing machine.

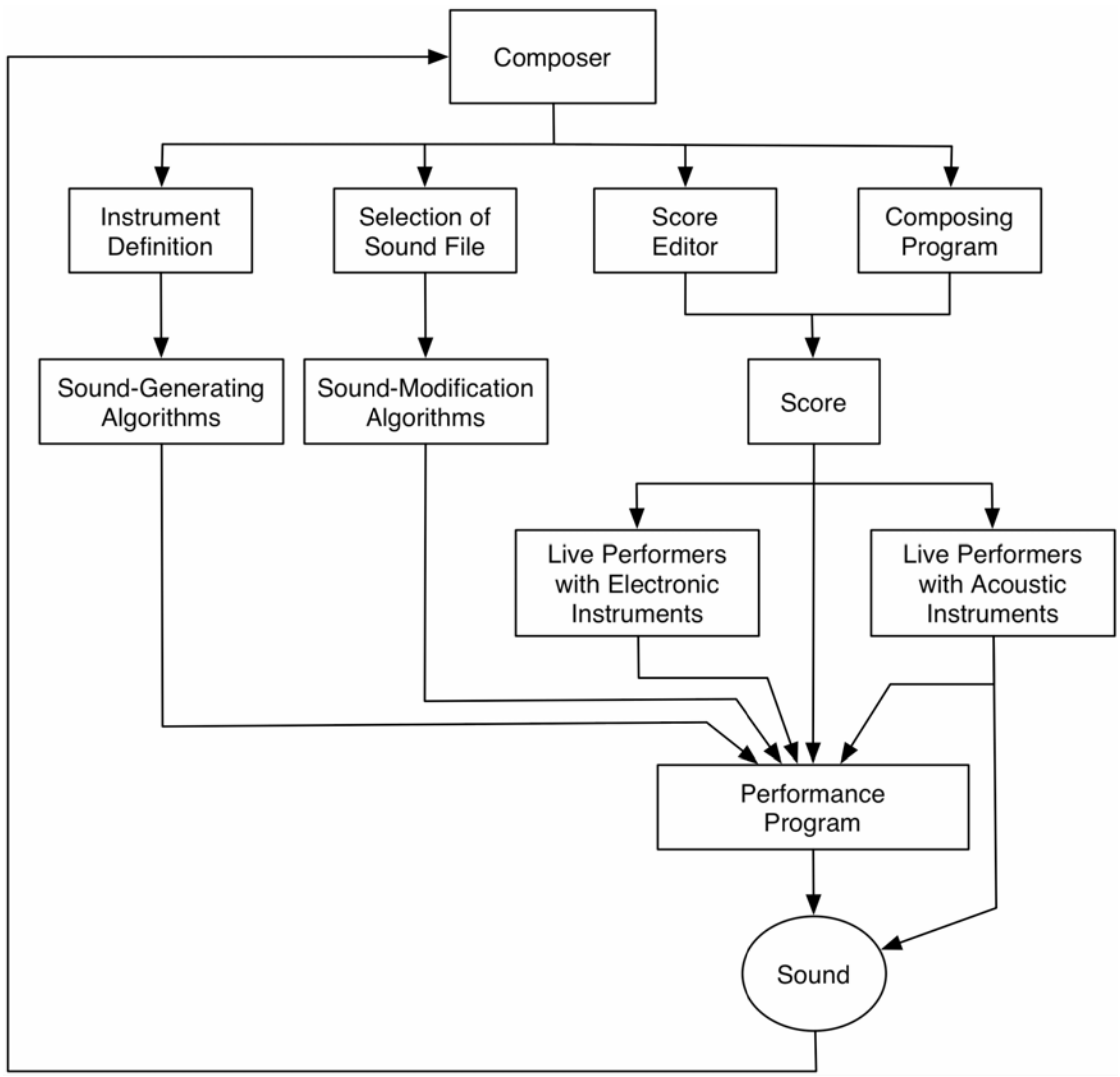


From Lejaren Hiller, Composing with Computers: A Progress Report, *Computer Music Journal*, 5:4 (Winter, 1981), pp. 7-21.

compositional thinking: music must first be composed, then it can be realized into sound.

Dodge and Jerse (1997) divided computer music tasks into four categories: "[1] algorithms for sound synthesis, [2] algorithms for modification of synthesized or sampled sound, [3] programs to assist the musician in composing with sound…and [4] programs that enable a computer performance of a composition" (p. 15). Like Hiller, they also created a model to demonstrate how a composer might use these tasks to compose music (see Figure 2). Dodge and Jerse's model was enhanced by their inclusion of the composer, which allowed them to clearly visualize the specific paths that lead to the ultimate goal of producing sound. Paths leading to sound synthesis and modification were given parity with paths leading to more traditional composition tasks such as score production and editing. This model was therefore a better reflection of Truax's fundamental trait of electroacoustic music composition than Hiller's model. Hiller's "realization" stage was replaced in this model by a "performance program" that serves to coordinate the other processes and render the composer's intention into sound. The model at this point became recursive, with sound returning to the composer for evaluation. The composer would presumably then use his or her judgment of the sound to determine further courses of action along the available paths.

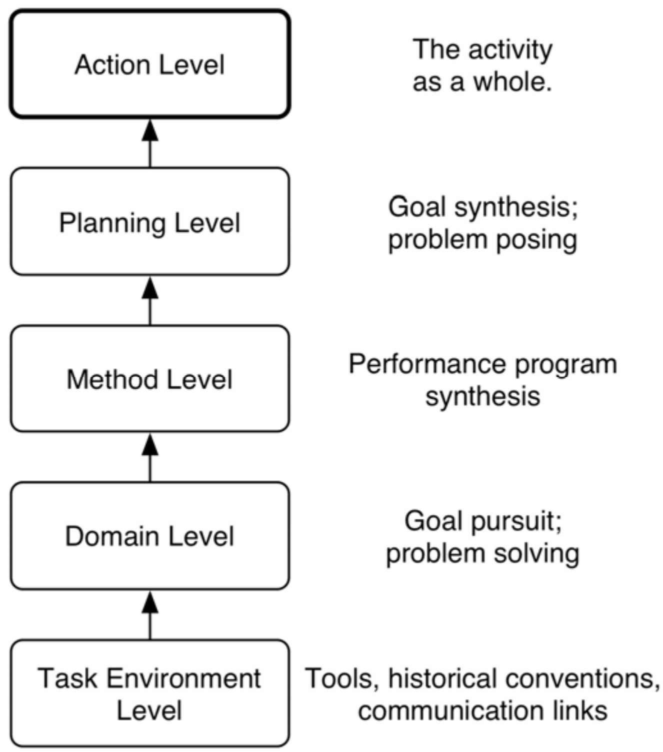
Figure 2. Some of the ways in which composers make use of computers.



Neither the Hiller nor the Dodge and Jerse models made any attempt to separate the activities of composition and programming from one another. Although the two activities may be interchangeable for the experimental composer, there is cause to consider the interaction between them. Lyon (2002) pointed out, "the experiments of today will lead inevitably to the norms of tomorrow" (p. 13). Experimental programs originally written for a single user have the potential to become normative tools for multiple users. Failure to consider the interaction between composition and programming can result in misconceptions about the influence such programs exert once the normative progression is complete.

Laske has written about "composition theory", which he described as concerned with "the process that underlies the design and realization of musical compositions" (Laske, 1989a, p. 119). As part of his research, Laske (1988) created a model of musical activity that divided processes involved into hierarchic levels (see Figure 3). The first level in his model was the "task environment" that included "tools without which the activity [was] unthinkable" (p. 52). In the case of a computer music composition, this level would include both the computer and the program used to create the composition. Laske (1989b) expressed interest in computer music because it "has given composers a tool for capturing their [compositional] processes" (p. 46) so that they may be evaluated later by others. He viewed the program resulting from experimental computer music composition

Figure 3. Levels of musical activity.



as an artifact that would document the compositional thinking underlying the final piece. Laske wrote from the viewpoint of a musicologist and found great promise in the potential for a program to capture portions of the compositional process.

Lansky (1990) offered his observations on the social changes caused by the introduction of computers into the music-making process. He began with the popular composer-performer-listener network as a point of reference and augmented the network with two additional nodes: the "sound-giver" and "instrument-builder." The sound-giver captured the change from live performance to recorded media as the primary means for experiencing music. The instrument-builder was not an entirely new node on the social network, because traditional musical instruments have been designed and built by dedicated individuals for thousands of years. However, Lansky attributed a new level of creativity to this node in recent decades because of computers and technology. He explained, "The vision of the instrument-builder can be idiosyncratic, and even compositional. Playing someone else's instruments becomes a form of playing someone else's composition" (¶ 16).

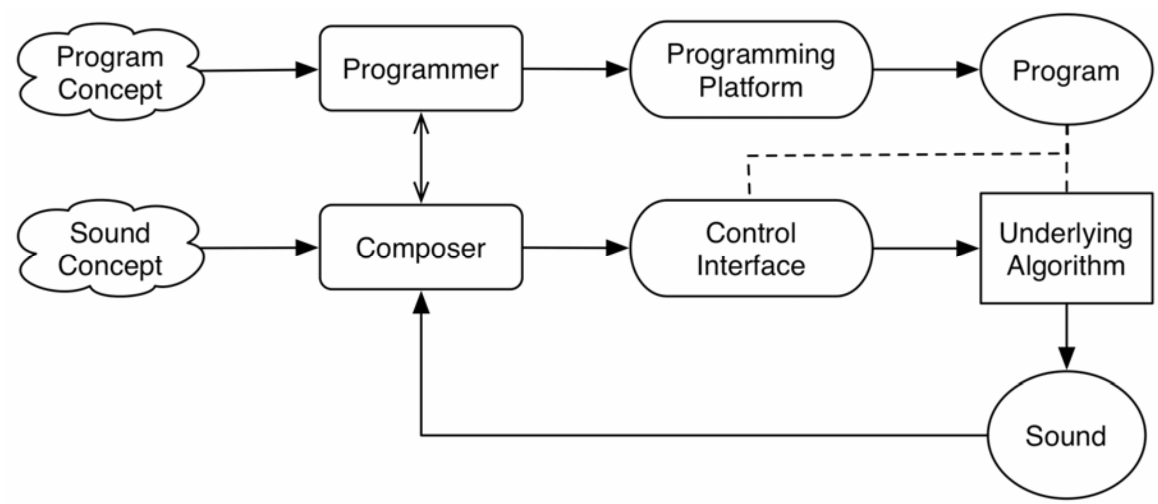
Both Laske and Lansky recognized the potential for programs to capture elements of the compositional thought process. However, Lansky's statement acknowledged the normative progression that enables the compositional formalisms contained within a computer music program to influence the composition of another person. Such reasoning opens the door to a kind of dual authorship shared by the programmer and composer and leads to a related question: whose musical contribution is primary? Truax (1986a) concluded that the answer was somewhere in the middle and stated, "a computer music composition reflects the musical knowledge embodied in both the software system that produced it and the mind of the composer who guided its realization" (p. 159).

B. A Model of the Programmer-Composer Relationship

In order to clarify the programmer's musical contribution when producing normative tools for computer music composition, it is useful to consider a model of the software design process. A new model proposed by the author is described in this section (see Figure 4). The model was designed specifically to reflect development of programs for sound synthesis or sound modification, a group of musical tasks that would include granular processing. The model may be applicable to programs for other musical tasks, however they are not considered here.

The model incorporates the two roles already discussed: programmer and composer. The potential exists for communication between these two roles. Experimental computer music can be viewed as a special case where both roles are embodied in a single person, providing an example with very good communication quality. The quality in other situations will typically depend on the

Figure 4. A model for the development and use of computer music programs.



specific circumstances, but will always offer less than the immediacy available to experimental computer music practitioners. In the case of software that is offered to users commercially, the communication quality is largely dependent on the size and business philosophy of the company responsible for the software's distribution.

The actions of the programmer begin with a "program concept" that motivates his or her actions. The program concept is an intentionally broad term that encompasses anything that guides a programmer while realizing the program, including wholly original ideas as well as attempts to realize specifications provided by another individual. There can also be multiple factors that are combined to form a new amalgam of these sources. Chowning (1996) stated, "while the function [that] a program is to perform can influence the choice of language in which the program is written, it is also true that a programming language can influence the conception of a program’s function" (p. xii). Chowning's observation is extended in the current model to the entire "programming platform," a term that encompasses both the *computer platform* and the *programming language* used. The elements are configured so that the programmer must act through the programming platform to create a program. The computer platform includes the identifying characteristics of a specific computer including the manufacturer, model and operating system. It most typically restricts the programmer in the form of available processing power. The programming language provides a means for the programmer to direct the operations of the computer. It may be a general-purpose programming language such as C (Kernigham & Ritchie, 1988) or a more specialized programming language for computer music such as Csound (Vercoe, 1986/1997; Boulanger, 2000) or Max (Puckette, 1988, 1991, 2002).

In order to capture the observation that programming is similar to composing (Chowning, 1996), the model contains analogous structures to describe the actions of the programmer and composer. The composer's actions begin with a "sound concept." As the program concept does for the programmer, the sound concept guides the composer's actions, including the selection of a program. The program can be considered to have two elements: a "control interface" and an "underlying algorithm." The control interface provides the user with the means for directing the actions taken by the underlying algorithm. The underlying algorithm defines the necessary steps for the computer to produce the desired sound output. The program must also define how directions received via the control interface will affect the underlying algorithm. In essence, the interface restricts the composer's actions in much the same way as the programming platform restricts the programmer's actions. Beyond this point the model breaks from its parallelism. The underlying algorithm produces sound as its output and this returns to the composer for evaluation as it did in the Dodge and Jerse (1997) model. In this respect, the composer is given a feedback mechanism that the programmer lacks. However, the author believes it is proper that only the composer perform the relevant evaluation.

C. Application of the Model to the Current Study

The programmer's potential influence as the instrument-builder (Lansky, 1990) is largely concentrated in the interface design. The programmer may choose to restrict an interface so that it controls only specific portions of the underlying algorithm, possibly leaving the other elements static or automating them via random processes. Such decisions are typically driven by the programmer's own plans for using the software, but the resulting limitations can likewise constrain composers' actions once the program develops into a normative tool. McNabb (1986) pointed out that when confronted by such a program, composers will spend "most of [their] time trying to figure out ways around the omissions and limitations of the design" (p. 150). To mitigate their influence upon composers and create programs that are free from omissions and limitations, programmers have often used a design strategy that Zicarelli (2002) called the "confrontational approach." He explained that this approach is identified by an interface that offers "complete explorations of a particular compositional approach… [and confronted] composers with controls and opportunities they might not ordinarily try" (p. 44). However, the mass of options comes at the expense of clarity, contributing to the complexity many new users attribute to confrontational programs and increasing the amount of time required to become proficient in using the program.

The majority of existing programs designed to produce granular processing sounds have exhibited the confrontational approach to design. These programs had interfaces that were saturated with controls for every aspect of the underlying algorithm. The current study begins with the hypothesis that the confrontational approach to interface design is largely responsible for the "amount of training [needed] to learn how operations in the micro domain translate to acoustic perceptions on higher levels" (Roads, 2001, p. 26). In their attempt to offer complete control over the underlying algorithm, programmers have neglected a full exploration of the potential interest in perception-based controls as an alternative. While psychoacoustic research has been an important resource for specific decisions regarding interface refinements, continued application would be enhanced through direct study of the cognitive response to specific granular processing examples. The findings of this study will become the basis for a new interface design that controls more perceptually relevant features. The resulting program should offer a simpler interface than a confrontational design by incorporating prudent limitations based on empirical evidence.

CHAPTER TWO: A GRANULAR PROCESSING PRIMER

Granular processing is part of a group of audio modification techniques united by a common foundational concept. Roads (2001) called this concept "microsound" and offered a comprehensive history of its development from antiquity to current applications in computer music. Such detail is beyond the scope of this document, however a brief account of the concept's modern development is offered in this chapter to place granular processing in its context. The author will then explain the basic methods underlying the granular processing technique itself and key terms that are used throughout the remainder of this document.

A. The Development of Granular Processing

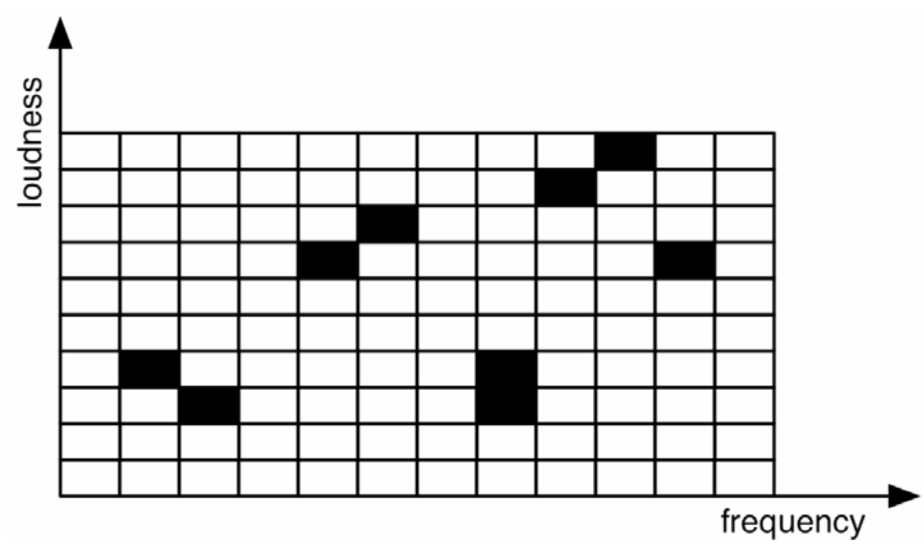
In *On Sensations of Tone* (1877/1912), Helmholtz laid the foundation for the classical model of human timbre perception. He stated, "the human ear is capable, under certain conditions, of separating the musical tone produced by a single musical instrument, into a series of simple tones, namely, the prime partial tone, and the various upper partial tones" (p. 25). By blending acoustics, physiology and music theory, he presented evidence that our ears are well adapted for detecting changes in the relationships between these component tones, giving rise to timbre perception. Although later research would point to the importance of other factors in timbre perception (see Risset & Wessel, 1999), Helmhotz's treatise remains a prominent part of the literature on this topic.

Gabor (1947) took exception to the tone as the fundamental constituent for complex sounds and claimed Helmholtz did not account for the effects of duration. He asserted, "it is our most elementary experience that sound has a time pattern as well as a frequency pattern" (p. 591) and therefore concluded that a fundamental sound unit should express both. He proposed the "acoustical quanta" as an alternative to the "timeless description" (p. 591) offered by most spectral analyses. Gabor described acoustical quanta as small bursts of "harmonic oscillations" (i.e., a sine wave) individually contained within an amplitude envelope, the shape of which would be based on a Gaussian distribution curve. According to his theory, these small sound "particles" were the universal basis for forming larger, more complex sounds. In other words, "any arbitrary signal can be expanded in terms of them" (p. 592).

In his piece *Analogique A-B (1958-9)*, composer Iannis Xenakis was the first to apply Gabor's particle concept to music. Xenakis (1963/1992) has referred to the two parts of this piece as separate "applications" of a single approach, calling them "*Analogique A*, for string orchestra, and *Analogique B*, for sinusoidal sounds" (p. 79). It was for the second of these that Xenakis created new timbres using "grains", his preferred term for Gabor's acoustical quanta concept. He concluded that in order to produce an aggregate complex sound, determining the characteristics of each component grain would require "months of calculations and graphs" (p. 49). In order to more effectively organize the grains at a higher level, Xenakis developed a system of grids with frequency represented on the x-axis and loudness represented on the y-axis (see Figure 5). Spaces on the grid were filled to represent activity at that frequency-loudness pair. Xenakis called these grids "screens" and organized them into sequences called "books". The sequences were used to describe the general development of grains through time, with stochastic processes filling in specific details at lower levels. Xenakis expressed excitement that his compositional approach to grains could create sounds that were "unparalleled and unimaginable" (p. 47) and used it to organize not only grains for *Analogique B*, but also the notes for *Analogique A*.

Roads (1978) made the first report of a computer program for *granular synthesis*. His method mirrored that of Xenakis: describe the high-level organization of grains and rely on stochastic processes to determine low-level details. His motivation was primarily efficiency, so that "by specifying several parameters at a higher level, a composer can call for the automatic computation of thousands of grains" (p. 61). Roads' primary program generated a score that was fed to a secondary program, an instrument created in the MUSIC V audio synthesis language to realize the score. His separation between the score and instrument programs exemplifies Hiller's (1989) division of computer music tasks

Figure 5. Screen like those used by Xenakis for describing granular activity.



that was outlined in chapter 1. This two-step approach to granular synthesis has since been be mimicked by other programmers (Helmuth, 1991, 1993; Nelson 2000a, 2000b; Orton, Hunt & Kirk, 1991).

Truax (1986b) developed the first program to allow the realization of granular synthesis in real-time. He accomplished this by interfacing a computer with a dedicated digital signal processor capable of continuously producing grains. Changes in the software interface immediately affected messages sent by the computer to control this hardware, so that the audible grains would follow the current settings. The immediacy with which the composer's actions affected the output blurred Hiller's task distinction and provided the possibility of composing the sound with direct audio feedback, as in the Dodge and Jerse (1997) model. The efficiency of the system was remarkable; it achieved grain densities that could "range up to 2375 gps ['grains per seconds']" (p. 231). Truax (1987) explained why the technique intrigued him in the following way:

> The fundamental paradox of granular synthesis – that the enormously rich and powerful textures it produces result from its being based on the most "trivial" grains of sound – suggested a metaphoric relation to the river whose power is based on the accumulation of countless "powerless" droplets of water. (p. 145)

Truax used the program to create all of the sounds for his composition *Riverrun (1986)*, a title inspired by his metaphoric impressions.

Both Roads (1985) and Truax (1987) separately extended their programs beyond the use of synthesized grains. Each began experimenting with audio sampled from an acoustic source as the basis for their grains, leading to the development of *granular processing*. Sometimes given a different name by individual programmers, such as "granular sampling" (Lippe, 1994) or "granulation" (Truax, 1987), granular processing is capable of producing a wide range of effects. Lippe (1994) observed that, "using granular techniques on sampled sounds offers a level of musical implication which does not exist in granular synthesis: one is acting on and transforming a pre-existing sound object" (p. 151). Two of the more straightforward transformations that granular processing can perform are time compression (Jones and Park, 1988) and time expansion (Truax, 1990), enabling the composer to arbitrarily shorten or lengthen a sound's duration. Both of these time alterations may be performed independently of modifications to the source sound's original pitch. Other transformations include the use of granular processing to create "continuous textures" that bear "no resemblance to instrumental and other note-based music" (Truax, 1987, p. 144). If this is true, they may engage human auditory perception in ways that have not previously been studied. In the current study, experiments were designed specifically to acquire a better understanding of how these continuous granular processing textures are perceived.

B. How Granular Processing Works

Gabor's particle concept has been used as the basis for many different computer music techniques beyond granular synthesis and granular processing (see Roads, 2001, chapter 4). Because of these diverse applications, it is necessary to establish an appropriate level of specificity prior to a discussion of this topic. When referring to all forms of sound synthesis or modification based on the use of sound particles, the term "microsound techniques" will be used in this document. The names of specific techniques, including granular processing, will be used only when appropriate. The term "granular techniques" will be reserved for references to granular synthesis and granular processing collectively.

Granular processing, like all other microsound techniques, uses short duration sounds as a basic component for creating more complex sounds. Unlike most other microsound techniques, the grains used for granular processing are sampled from a digital audio recording of an existing sound source (see Figure 6). An *amplitude envelope* (see Figure 7) is applied to a short segment from the source sound to create a fade-in at the beginning and fade-out at the end, forming a single grain of sound (see Figure 8). Without the envelope, the sound segment could potentially start or stop too suddenly, producing an audible click caused by sample discontinuity. Scaling the envelope to a specified

Figure 6. Approximately 20 ms from a recording of a handbell.

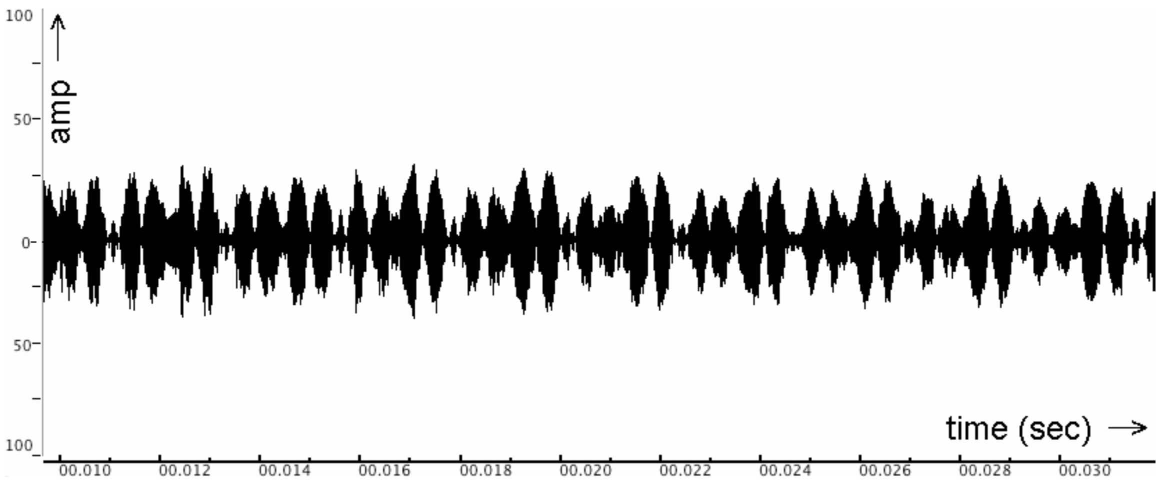


Figure 7. Gaussian amplitude envelope.

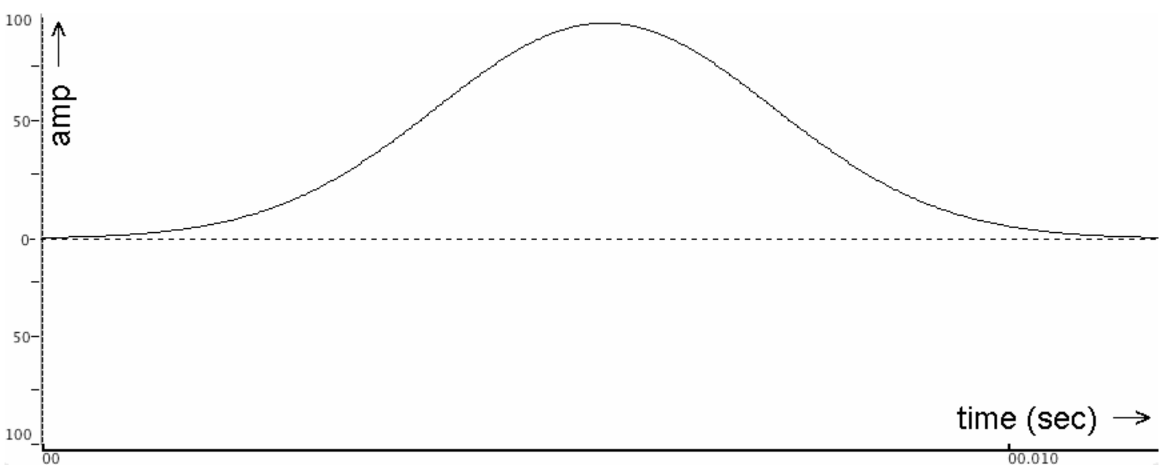
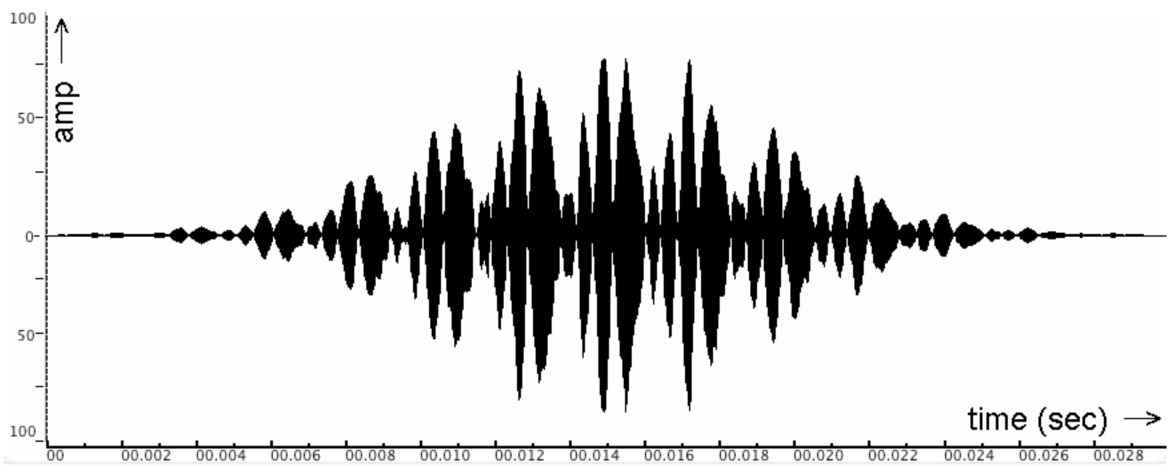


Figure 8. Single grain combining the sound source in Figure 6 and the amplitude envelope in Figure 7.



level can also act as a means to limit the maximum amplitude of a grain, providing control over its perceived loudness.

The most common *window shape* or *windowing function* used for these amplitude envelopes is based on the Gaussian normal distribution curve (see Figure 7). This shape is historically significant because Gabor (1947) proposed it as part of his original conception of grains. Computer programs for granular processing have used other window shapes, including modified forms of the Gaussian (Roads, 1978) and trapezoidal windows (Truax, 1988). Truax chose trapezoidal windows primarily because of the ease with which they could be computed, thereby reducing the processing load for his real-time program. Keller and Rolfe (1998) have since studied the spectral effects induced by a trapezoidal window shape and offered reasons why its effects may make it desirable. Some of these alternative shapes provide the composer with separate control over the *attack* and *decay* portions of the envelope (see Figure 9). Increasing or decreasing the duration of either would alter the overall shape of the envelope and create corresponding changes in the sound of the grain.

The *grain duration* or *grain length* describes the amount of time from the start to finish of a grain (see Figure 10). Most granular techniques work with grains that are on the order of 10 to 50 ms in length. To put this in perspective: Sixteenth notes last approximately 100 ms at a metronome marking of 150 beats per minute. The tempo would have to be twice as fast for the beats to approach

Figure 9. Trapezoidal amplitude envelope with the attack and decay labeled.

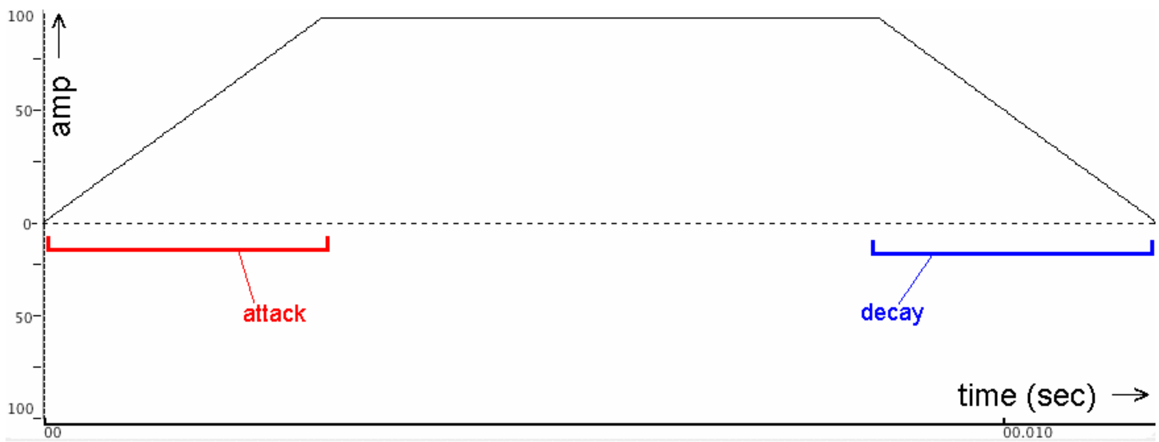
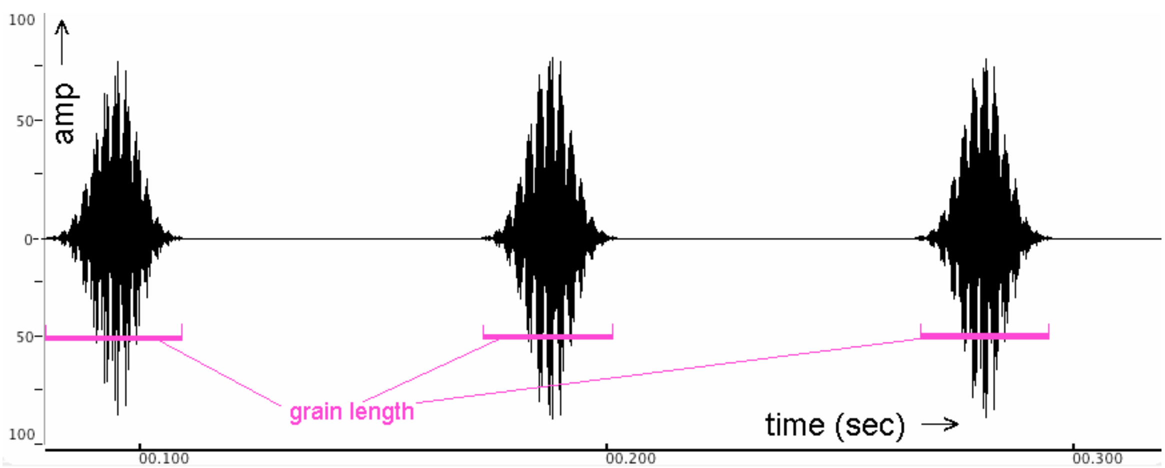


Figure 10. Grain length or duration.



the upper limit of grain durations. It is because the grains are short and occur in such rapid succession that they are not perceived as isolated events and instead form their characteristic "rich and powerful *textures* [italics added]" (Truax, 1986b).

The 10 ms lower limit can be traced directly back to Gabor (1947), who cited a 1935 study by Bürck, Kotowski and Lichte as justification for the number. Their study showed 10 ms as the "minimum duration" at which oscillators could be "heard as a short musical note, with ascertainable frequency" (p. 592). At shorter durations, listeners would hear only a click. Later studies conducted separately by Meyer-Eppler and Olson (both cited in Butler, 1992) confirm this phenomenon, but found the minimum to be dependent on the frequency used. The minimum duration for the highest frequencies tested was 13 ms, while lower frequencies required 45 ms to be perceived as pitched. Longer durations tend to be more desirable for granular processing due to the influence of the sampled audio. Truax (1994) explained, longer durations ensure that "the timbral character of the original material is the least modified," when compared to the spectral effects "created by shorter grains that extend the sound's bandwidth" (p. 40). However, it would be undesirable to use durations beyond the range's upper limit because "grains with durations much longer than 50 msec tend to be perceived as separate events" (Truax, 1988, p. 18), rather than as a unified texture.

Higher organization levels for granular techniques have been labeled *events* (Roads, 1978) and *voices* (Truax, 1986b). An event consists of the overall texture from its start to finish created by thousands of grains sounding together with a measured span of time, whereas a voice is a single stream of consecutive grains. Adding and subtracting voices is one method used to thicken the overall texture of an event. The voice concept most likely arose as a programming strategy for real-time granular synthesis, so that messages could be routed more effectively to available resources. Because it is an effective means for controlling grain production, other programmers have used this method since (Eckel, Rocha-Iturbide & Becker, 1995; Lee, 2000; Rolfe & Keller, 2000; Tanghe, 2003; Todoroff, 1995).

Within a given voice, the *grain period* is the amount of time between consecutive grain starts (see Figure 11). It is analogous to inter-onset interval, a term often used in music perception literature to describe the rate at which notes are played. As an alternative, voices are sometimes described in terms of the *grain delay*, a measure of the time between grains in which a voice is silent (see Figure 12). "Pulsar synthesis" (Roads, 2001, p. 137), a related microsound technique, uses a single ratio to describe the relationship between the grain duration and grain period instead of two independent controls. Because of the variation that exists among programs (see chapter 3), it is unclear which of these descriptors represents the best control for this aspect of granular techniques.

Figure 11. Grain period or onset interval.

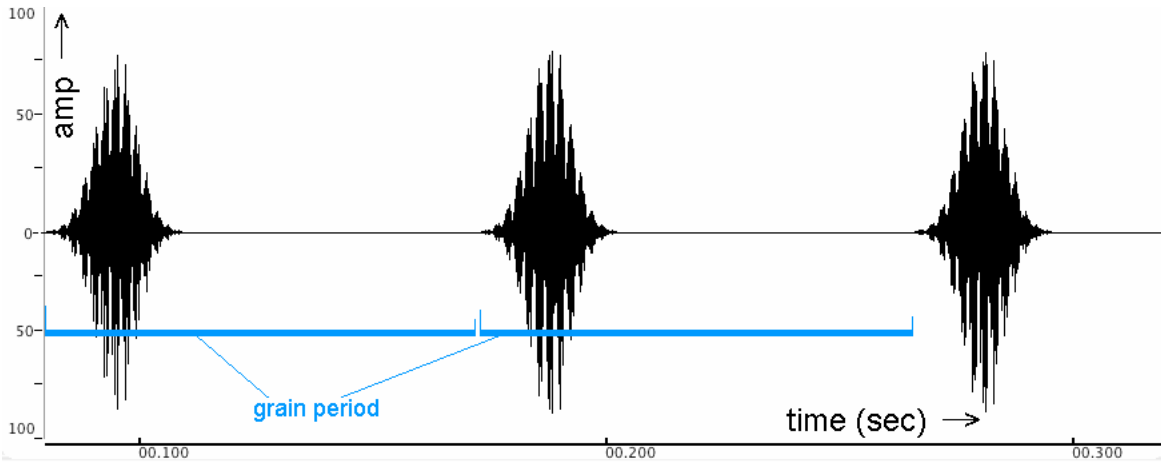
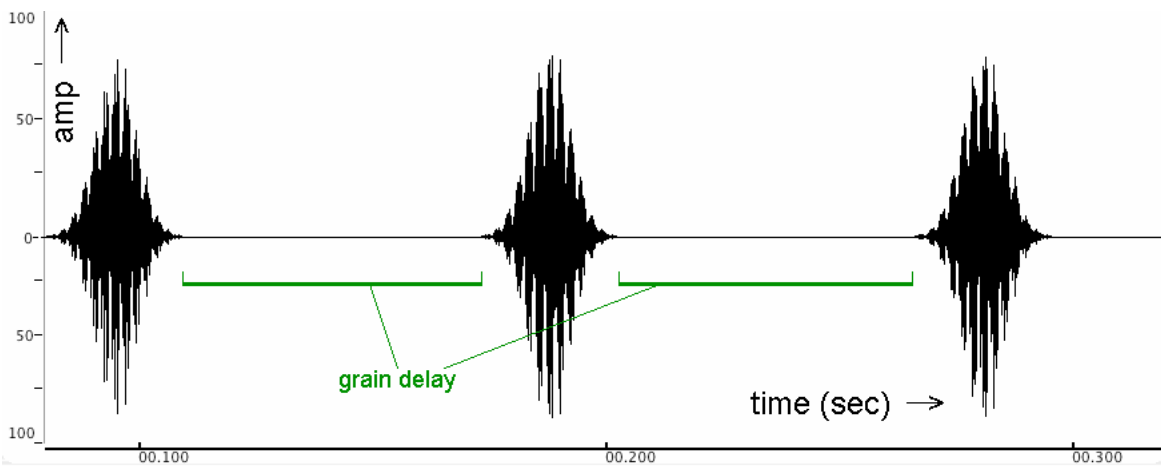


Figure 12. Grain delay.



The amount of time between grain onsets is typically randomized in granular techniques, thereby avoiding any regular period within a voice. Roads (1985) noted that without such randomization, granular techniques become functionally equivalent to another computer music technique known as *amplitude modulation*. Irregular grain periods are partly responsible for the characteristics associated with granular sounds. Because grain duration and grain period are varied independently, the period may occasionally be shorter than the duration causing consecutive grains to overlap and produce a smoother texture. Roads (1991) later categorized granular techniques using the criterion of whether grains overlap, calling those that did "asynchronous" and those that did not "quasi-synchronous".

Some granular programs use the term *density* to describe the number of grains per second (gps). For a single voice, density can be expressed as the reciprocal of the granular period expressed in seconds. It is also a measure of how frequently grains are produced and therefore the terms *grain frequency* or *grain rate* can also be applied. With a minimum duration of 10 ms, density can reach 100 gps within a single voice of non-overlapping grains. Multiple voices can be used within an event to increase the density. The earliest granular synthesis programs were capable of 1,600 (Roads, 1985, p. 151) and 2375 (Truax, 1986b, p. 231) gps respectively. Although higher densities are now possible with the increased speed of computers, as the density increases the sound output begins to mimic white noise, something that can be produced by far more efficient means.

It is the density of granular textures that makes specifying low-level parameters for each grain impractical for the composer, as alluded to in the previous section. Xenakis' solution, to rely on stochastic processes constrained by a high-level description of the overall texture, has been adapted and applied by programmers to all parameters for granular techniques. The relative ease with which pseudo-random number generators can be incorporated into a computer program makes them very efficient. However, there are differences in how programmers have controlled these random values. Some have constrained values within a set of maximum and minimum values for the parameter (Roads & Alexander, 1997; Tanghe, 2003), whereas others have defined a mean value and its bandwidth or deviation (the difference between the maximum and minimum values; Roads, 1978; Truax, 1988). A few examples exist where the programmer has used a combination of the two (Helmuth, 1991; Wolek, 2001).

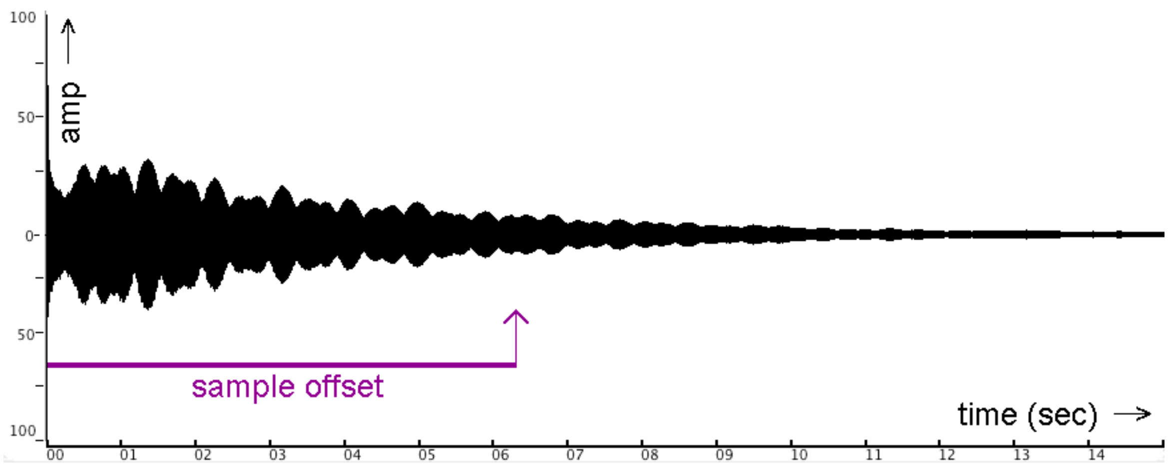
Descriptions for these randomization constraints have also differed. Some programs express the amount of deviation in the same units used to express the mean (i.e., a grain period of 40 ms with a bandwidth of 10 ms; Rolfe & Keller, 2000), while others use a percentage or ratio (i.e., the same 40 ms grain period would have a 25% total deviation; Lee, 2000, Wolek, 2001). The choice of percentage is based on the assumption that the amount of random deviation should be dependent on the mean value, the same way that "Q" functions to define bandwidth relative to the center frequency of a bandpass filter. As the mean value is changed, bandwidth of these random deviations will change proportionally. For example, a uniform 20% deviation at mean values of 20 and 30 ms would create bandwidths of 4 and 6 ms respectively. Evidence to support a perceptual preference for one of these methods is absent from the literature and therefore choosing between them has been left to the discretion of individual programmers.

Granular processing also has parameters that control how samples are drawn from the audio source being transformed. The rate at which samples are read from the source for processing can be controlled via the *sampling increment*. Setting the sampling increment to one (1.0) will result in the audio being read at a normal rate. By up-sampling or down-sampling the digital audio source within a grain, the processing can alter the perceived pitch of the output. Whole numbers greater than one (e.g., 2, 3, 4, etc.) will cause the source to be transposed upward along the harmonic series, whereas fractional values can be used to create smaller interval transpositions (Truax, 1994). The parameter may also be varied between the constituent grains of a particular voice or event, so that multiple pitches would be heard as part of the texture. It is because of the perceptual results caused by changes to the sampling increment that parameters for its control have been referred to as "pitch" (Lippe, 1994; Wolek, 2001), "transposition" (Eckel, Rocha-Iturbide & Becker, 1995; Todoroff, 1995) and "harmonization" (Truax, 1994). This is a clear example of how interface design choices may be informed by auditory perception.

Finally, the segment of the source audio that is to be sampled for granular processing must be specified. This parameter is called the *sample offset* or *onset time* (see Figure 13) and is typically specified by the amount of time between the beginning of the audio source and where sampling should begin (Truax, 1987). Varying this parameter is a means of effecting the listener's perception of time. Lippe (1994) maintained that this granular processing parameter was a crucial determinant in the listener's ability to recognize the sound source and explained his reasoning in the following way:

> Using a piano note as the stored sound, if the onset times descend in an ordinal fashion from high to low, while the density distributions of all other parameters are randomly calculated, the sounding result will always be recognized as a piano note played backwards even though variants may sound quite different. Furthermore, the ability to "deconstruct" sounds via the manipulation of onset times in granular [processing], moving between the boundaries of recognizability and non-recognizability on a continuum, is one of the principle, musically interesting characteristics of granular [processing]. (p. 151)

Figure 13. Sample offset.



Controlling the rate at which the sample offset proceeds through the digital audio source produces the effects of time compression (Jones & Parks, 1988) and time expansion (Truax, 1990) that were previously mentioned. When the sample offset remains static for the entire duration of a granular voice or cloud, it results in an emphasis of the granular texture over the sound source.

C. Observations

The author of the current study is himself a programmer who has developed software for producing sounds based on granular techniques (Wolek, 2001). Because of the breadth of algorithm controls that must be considered, design questions were constantly raised during the development of this software. What parameters are really needed? What values are appropriate and should they be constrained to a certain range? How does randomization affect the way this sounds? While some questions were answered using the information gleaned from existing literature and methods that have been proven effective by other programmers, many questions were answered using my personal perception of sounds that the software would produce and their appropriateness for a given compositional context. The documentation and writings related to other programs for granular techniques reveal that this experience is not unique.

The current empirical study of the perception of granular sounds has the potential to provide insights that could lead to new answers for these design questions. Such answers may contradict or confirm those found in the past, but they will be based on quantifiable empirical evidence. Because granular processing has so many possible parameters, it was necessary to limit the focus of this initial study to a few parameters. Before selecting specific parameters, a thorough review of previous programs was undertaken. An account of this review will be given in the following chapter.

The creative adaptation of Gabor's original concept by composers and programmers has produced a variety of microsound techniques. Among these, granular processing has received special attention because of the manner in which it transforms a familiar sound source. This chapter has provided a concise introduction to granular processing's methods and terminology so that the previously uninitiated reader can more easily approach the current study.

CHAPTER THREE: REVIEW OF EXISTING PROGRAMS

DePoli and Piccialli (1991) noted, "In general, granular synthesis is not a single synthesis model but a way of realizing many different sound production models" (p. 187). Their statement expressed the popular view of granular techniques that has enabled each programmer to adapt and augment at will. Granular processing is itself an extension of the original granular synthesis model. Unique approaches abound within programs implementing granular techniques. Each seems to use different interface elements for controlling the sound production or idiosyncratic terminology for describing the technique. This chapter will examine specific computer applications that implement granular techniques, with a focus primarily on granular processing. Special attention will be given to the innovations introduced by individual applications. Examples that relate to the model of computer music programming presented in chapter 1 will be highlighted.

A. Software Examples

1. Music V Granular Score Generator (Roads, 1978)

As mentioned in chapter 1, Curtis Roads' 1978 report represents the beginning of modern granular synthesis. In describing his implementation, Roads wrote about the difficulty of specifying an individual set of control parameters for each grain within a dense grouping and the potential efficiency of automating grain production from a high level. While his software accomplished this efficiency, important control parameters were missing. For instance, the duration of his grains remained constant at 20 ms and a maximum of 32 grains could be produced at one time. The parameters his program offered the composer were as follows:

- beginning time and duration of event
- initial waveform and waveform rate-of-change (slope)
- initial center frequency and rate-of-change of center frequency
- initial bandwidth (frequency dispersion) and bandwidth rate-of-change
- initial grain density and grain density rate-of-change
- initial amplitude and amplitude rate-of-change (p. 61)

The program allowed composers to specify parameters as a group of initial values and determine how they would change over the course of the granular event. It relied on an instrument program developed with the MUSIC V synthesis language running on a Burroughs 6700 computer to render the resulting sound files. As was stated in chapter 2, the design of this score generation program fits Hiller's model of computer music tasks.

Roads stated early in his report, "Automated granular synthesis is a fruitful technique for the exploration of an entirely different class of computer-generated sound spectra than the usual…methods" (p. 61). However, he concluded by cautioning that granular synthesis "is not an 'all-purpose' synthesis technique" (p. 62), a comment that seeks to capture some of its limitations.

2. Music 11 Granular Score Generator (Roads, 1985)

In addition to providing a thorough account of the theories behind granular synthesis, Roads (1985) described efforts to update and extend his previous work. He created an updated instrument program with the Music 11 synthesis language running on a DEC PDP-11/50 computer. Using the same two-step model, this instrument program again relied on instructions generated by a score program based on composer-provided high-level parameters. This revised implementation included a new ability to vary individual grain durations. Roads reported that grain durations down to 10 ms can be "quite effective," but that below this threshold grains become "computationally costly" (p. 150). The increased strain on computational resources provides a second reason for the grain duration to remain above 10 ms (in addition to the perceptual considerations mentioned in chapter 2).

Roads also reported the first testing of granular processing, referring to the process as "time-granulation". He claimed to have used granular processing on "many soundfiles, including the sounds of snare drums, cymbals, and tom-toms," as well as a saxophone. He manipulated "the duration of individual grains, the density of grains within an event, and the number and type of soundfiles used as input to the granulating instruments [to make] a wide range of textures achievable" (p. 158).

3. GSAMX (Truax, 1988)

A composer using the GSX and GSAMX software created by Barry Truax could hear the results of parameter changes immediately via the continuous audio output. The software ran on a PDP-11 computer interfaced with a DMX-1000 dedicated digital signal processing (DSP) hardware. Truax (1988) provided a summation and extension of two papers he had previously presented at the International Computer Music Conference (Truax, 1986, 1987) about these programs.

Of the two programs, *GSAMX* was the one used for granular processing. The parameters supported by this program are as follows:

- offset number of samples from the start and offset range…
- average grain duration and duration range
- delay time between grains…
- speed of output, which acts as a pitch/time transposition
- number of voices sounding at transposed sample rates
- total number of voices sounding (maximum = 20) (p. 17)

Changes in these parameters would cause the program to send corresponding commands to the dedicated processing hardware.

Some limitations of Truax's hardware system had important effects on the sound output. Commands were sent to the DSP hardware every 1 ms, a limitation that affected the ability of GSAMX to make smooth transitions during some parameter changes. This timing limitation restricted values for those parameters involving time (i.e., grain duration and grain delay) to millisecond increments. Only 4 kilobytes of computer memory were available to store digital audio for sampling, which allowed it to hold 150-170 ms of sound source. Truax used this buffer in two ways: first, as a fixed buffer from which grains could be sampled and second, "acting as a short delay-line or time window that is tapped to furnish the various grains" (p. 17). Both options were restricted to operating upon the same limited amount of sound source at any given moment. These were specific examples of hardware limitations that constrained the programmer's ability to implement his program concept.

After describing the control variables for his software, Truax offered observations on the psychoacoustic implications of granular techniques, a topic of particular interest for the present study. He stated, "Each of the control variables cited previously have [sic] a psychoacoustic correlate that may be more suggestive as a basis for compositional organization than the numerical values of each variable" (p. 18). His views about these correlates are relevant to the current study. First, Truax related his impressions of the perceived pitch of granular synthesis textures, observations that are not applicable to granular processing. Second, he offered comments on the effects of duration and delay, both relevant to all granular techniques. He connected these parameters to the idea of density, or the number of grains that sound within a given second. He cites 50 ms as a threshold beyond which the grains begin to sound less like a unified texture and more like "discrete events – a kind of 'pulling apart' of the component grains" (p. 18). According to Truax, density has an inverse relationship to the duration and delay parameters, leading to the perception of less density as these two values increase. Lastly, Truax connected the randomization of duration and delay to the concept of periodic modulation. He explained that without randomization, granular techniques produce "an amplitude-modulated signal" (p. 17), something already noted by Roads (1985). As the amount of randomization decreases, the periodicity increases and vice versa.

Although he recognized the indirect relationship between his user controls and listener perception, Truax did nothing to adapt his program so that it provided more perceptually relevant controls for granular processing. Of course, Truax had no empirical evidence to support his "perceptual correlates," only personal observations. His work on the GSAMX program has influenced many others since it was published and may have sent future software development in a very different direction had he acted on his intuitions.

4. CMIX Pitch Shifting Algorithm (Lent, 1989)

Lent (1989) downplayed the similarities between his pitch shifting algorithm and granular processing, admitting only that they both sample portions of a source sound, apply windows and recombine them to produce transformations. He insisted, "the similarity between these two methods ends here" (p. 70). But Keith Lent's underlying algorithm to achieve pitch and time transformations was an example of how a granular processing program could operate at a higher level than granular events. The parameters given to the composer included the following:

- Nin – number of input samples
- IN(n) … – the array of input samples
- Nout – number of output samples
- Periodratio – factor by which the period length should be altered; i.e., the reciprocal of the frequency shift ratio
- OUT(n) … – the array of pitch shifted output samples (p. 68)

The composer was not required to specify the parameters seen in previous examples (e.g., grain duration, delay, and randomization). These parameters were computed automatically for the underlying algorithm after the program performed an analysis of the input signal.

Offsets into the source signal were computed so as to result in phase-alignment between consecutive grains. Put more simply, as one grain would begin to fade out, the analysis searched through the source to find a point on the waveform that had the same amplitude value as the one being output by the current grain. The program then used this new sample position as the next offset value for a second grain that would begin to fade in. This type of phase alignment is required for more transparent pitch shifting results. Because a limited set of control parameters were provided, this application of granular processing offered limited output possibilities. However, by focusing his desired effect, Lent was able to implement a concise list of parameters that provided a clear connection with desired results.

5. Time Expansion with GSAMX (Truax, 1990)

Truax (1990) offered another example of the transformations composers could achieve with his GSAMX software. He explained how to use his software to stretch playback time without influencing the perceived pitch of a sound source. The technique centered on the rate of progression for the sample offset through the source audio and its relation to the normal sampling rate used for playback. Truax explained, "the rate of the time-shifted sound is defined as the ratio of 'off' milliseconds to 'on' milliseconds and is called the 'off:on ratio'" (p. 105). This ratio affected the way samples were read into the memory buffer and used for granular processing. During the "on time", sound was read into the buffer for the specified number of milliseconds at the proper sampling rate; during the "off time", the buffer was held unchanged. These two processes alternated continuously, resulting in the overall effect of the source being expanded or stretched over time.

Just as he did previously, Truax recognized the disconnect between his parameters and the perception of the resulting granular transformation. He proposed the "Time Extension Factor (TEF)" (p. 106) as a better descriptor of his time expansion effect, and offered the following formula for its computation:

Time Extension Factor = (Off Ratio + On Ratio) / (On Ratio)

There was however the potential for multiple on:off ratios to result in the same TEF value (e.g., off:on = 1:1 vs. off:on = 5:5; in both cases TEF = 2), an issue that Truax never addressed.

6. StochGran (Helmuth, 1991)

Mara Helmuth (1991) described her *StochGran* program for granular synthesis, a program that allowed composers to use a GUI for controlling a set of high-level parameters. The program created a score output for an instrument program built in the CMIX music synthesis language. In addition to typical controls over the source sound, amplitude envelope and sample offset, Helmuth provided the composer with controls over the randomization of density, duration, location and pitch.

Helmuth's implementation of randomization was unique. Instead of constraining random values by the maximum-minimum or mean-bandwidth definitions (see chapter 2), *StochGran* provided the composer with a kind of hybrid of the two. Helmuth explained, "The user can specify the mid point and the low and high boundaries of the parameters chosen. Also a 'tightness' around the midpoint is chosen. A simple probability function will generate values within these limits which hug the midpoint with the desired degree of tightness" (p. 564). The ability to distribute random values in a non-uniform manner provided the composer with more sound output possibilities, but it also lead to an interface with over 40 parameters. Helmuth would extend her contributions to granular research in later years (Helmuth, 1993; Helmuth & Ibrahim, 1995), but this non-uniform distribution was perhaps her most noteworthy addition to modern granular techniques.

7. Granular Sampling for Max on the ISPW (Lippe, 1994)

Cort Lippe created a granular processing patch using Max (Puckette, 1988, 1991, 2002) on the IRCAM Signal Processing Workstation (ISPW; Lindeman, Starkier, & Dechelle, 1990). Max is a graphical programming environment that allows the programmer to work by connecting objects. Each object produces output based on input that it receives, enabling the user to easily experiment by "patching" together various configurations of connected objects. Max has since been commercialized and is currently distributed on Mac OS X and Windows XP (Cycling74, 2001).

Lippe (1994) distinguished between granular synthesis and what he called "granular sampling" (p. 150), pointing out that his software was intended for the latter. Although the name was different, this technique was equivalent to granular processing. He discussed the possibility of some sampling parameters being more perceptually important than others, specifically in relation to the sample offset. The manipulation of sample offset (or "onset time" as he called it; p. 150) contributed to a higher-level concept of "*grain order*" (p. 151) that could reveal the source to a listener. Because of conventions in the Max architecture, implementing controls over grain order was left to the user even though Lippe insisted on its perceptual importance. His implementation of granular sampling included parameters for "onset time into sampled sound, pitch, envelope description, maximum amplitude, grain duration, rate of grain production, overlap of grains, and spatial location of each grain," with all of these "controllable in real time" (p. 151).

In addition to randomizing each parameter as others have done, the patching paradigm of Max lead Lippe to experiment with the idea of mapping audio analysis to control parameters. He described his use of this technique to compose pieces for acoustic instruments accompanied by the ISPW. Lippe explained, "Continuous pitch and amplitude tracking of a performance offers musically relevant data which can be used to control aspects of an electronic score, and perceptually create coherence between the instrument and electronics" (p. 154). It was an interesting technique that could certainly yield new possibilities, but beyond the scope of this document since we are interested in building an interface for direct control instead of the acoustic analysis mapping strategies to which Lippe referred.

8. GiST for Max/FTS on the ISPW (Eckel, Rocha-Iturbide, & Becker, 1995)

GiST was a group of objects for the Max environment that were designed to help alleviate "the lack of precise temporal control" (Eckel, Rocha-Iturbide, & Becker, 1995, p. 299) that the authors identified within Lippe (1994). Because control messages on the ISPW were only exchanged at intervals equal to 64 samples of audio output, grains could only be triggered at this interval or at integer multiples. Their solution was to develop new objects that used a specialized message format. These messages did not immediately trigger a grain upon their receipt. Instead, they specified an amount of time for the object to delay between the message's arrival and a grain's beginning. This new strategy allowed the number of grains per second to be more accurately controlled by the composer.

Grain production was handled by an object called FOG that provided the user with several control parameters (e.g., source sound, sample offset, a linear transposition factor, frequency bandwidth expressed in Hertz, and amplitude). The composer could not specify the overall duration of grains, only its constituent parts. The amplitude envelope for each grain was controlled through independent attack, sustain and decay durations, each expressed in milliseconds. However, a composer could easily adapt to this requirement by using the Max environment to build higher-level controls that would output the necessary lower-level parameters for the FOG object.

9. Gesture Control within Max/FTS (Todoroff, 1995)

Todoroff (1995) provided the "usual control parameters" for granular processing, including "amplitude, attack, sustain and release time of an individual grain, delay between successive ones, [and] transposition factor" (p. 317). His software also had an option to synchronize two parallel granular voices, so that the attack time and duration of the main voice would be linked to the decay time and duration of its complementary voice. He reported this option to be "very effective for performing independently varying time-shifting and frequency transposition" (p. 317).

More importantly, Todoroff built interface controls that translated gestures into control parameters for his underlying algorithm and attempted to adjust for the non-linearities in how they would be perceived. As he explained, "changing the grain duration from 5 to 6 ms could have a tremendous effect on the sound quality, but no one would notice a change from 999 to 1000 ms" (p. 316). Todoroff moved beyond the recognition offered by his predecessors and implemented a more perceptually relevant control. However, he did not explain the specifics, nor did he cite any evidence to justify his mapping strategy.

10. Cloud Generator (Roads & Alexander, 1997)

Roads documented the *Cloud Generator* program for granular processing in his book *Microsound* (2001, appendix A). The application's name came from Roads' prior descriptions of granular events as "clouds of granular particles" (Roads, 1978, p. 62). The source sample used for cloud production could either be a predefined, synthesized waveform (e.g., a sine, sawtooth or square wave), a user-drawn waveform, or an imported sound file. It was the imported sound file option that enabled the program to perform granular processing. However, memory limitations prevented sound files longer than 46 ms in duration from being used, a limitation greater than that found in Truax (1988). Instead of direct control over the sample offset, the composer selected from one of three control methods: 1) totally random, 2) beginning-to-end order with some random deviation, or 3) a strictly beginning-to-end order. The use of three distinct settings instead of a continuum may reflect Roads' belief that finer distinctions could not be perceived.

A simple GUI was used to manipulate the remaining parameters. Some parameters remained fixed while others evolved over the course of the cloud event. These evolving parameters helped to give the audio output a dynamic quality as parameters progressed from their beginning values to ending values. These parameters included density, grain amplitude (expressed as a percentage of the maximum), grain duration, placement within the stereo field, and a high and low "bandlimit" (Roads, 2001, p. 386). Roads' bandlimit parameters controlled the pitch randomization using the maximum and minimum boundaries, notably different than his earlier implementation (Roads, 1978) that had constrained frequency randomization using the mean and bandwidth.

The bandlimit parameters were obviously geared toward use with the synthesized waveforms. When used with the sampled audio option "this waveform will repeat at its extracted frequency only if the cloud bandlimits are set to 21 Hz" (p. 387). The composer was required to compute any desired pitch manipulations based on their relationship to 21 Hz, extra work that could have been avoided by adapting the software interface when the sampling option is selected. This type of adaptation had been implemented for controlling grain duration. Duration had an added option to apply randomization, which if checked caused "the grain duration parameters [to] switch from 'Initial' to 'Minimum' and from 'Final' to 'Maximum'" (p. 386).

11. Stampede II (Behles, Starke, & Röbel, 1998)

Stampede II performed granular processing on either recorded sound files or live audio input. It ran on Silicon Graphics brand computers and won the 1997 Bourges International Software Competition. The processed audio output could be heard in real-time or recorded directly to disk. Several sampling modes were offered, affecting the program's analysis of the source sound before producing grains. Each had its own advantages and disadvantage, particularly related to the reduction of artifacts produced during processing. The description here will focus on the "quasi-synchronous" (p. 44) mode, because it was the most similar to methods found in the other profiled software.

There was some variation in the parameter names that may be attributed to the fact that the researchers responsible for *Stampede II* were not native English speakers. However, at least one of the name changes had an intentional meaning. The parameter for manipulating pitch was labeled as a "pitch-formant shift." The programmers reasoned that the method used to achieve this effect caused "the spectral envelopes [to be] shifted with the pitch, resulting in a timbral modification" (Behles, 1998). This was actually a more accurate description of the effect created by up-sampling or down-sampling the sound source for a grain.

Several parameters had to work in tandem to control the grain duration and amplitude envelope. Instead of a grain duration control, *Stampede II* included a parameter called "grain width", a ratio that related duration to the grain frequency. The grain duration in seconds would be expressed as the quotient of the grain width divided by the grain frequency. The "grain fade" parameter "[controlled] the shape of the grain envelope by determining the ratio of the fade-in and fade-out portion of the envelope to the total grain duration" (Behles, 1998). This was problematic because the interface did not provide direct control over the grain duration. Having conjoined controls over these fundamental parameters was a unique innovation by the programmers, but potentially confusing for those with prior granular processing experience.

The program interface was divided into "*primary parameters* and *secondary parameters*. The former determine mean values, and the latter control… deviations from the mean" (p. 49). The programmers used this principal as the basis for grouping primary and secondary parameters within columns (see Figure 14). The name of each primary parameter and its unit of measurement were located at the top of each column with slider controls placed beneath them. Secondary parameters were presented in a consistent manner down each column. The current value was displayed next to each slider and updated as the composer manipulated the interface via a mouse. Although organized into columns, the *Stampede II* interface still had over 40 parameters for the composer to control the sound output. This was a clear example of the confrontational approach to program design outlined in chapter 1.

Most primary parameters were grouped in their designated column with three secondary parameters for controlling modulation elements. The first of these controlled random deviations from the mean, similar to the bandwidth parameters found in the software already profiled. The second was called "voice-proportional modulation" and allowed the user to specify a value that would be applied in integer multiples to available granular voices, alternating between positive and negative values. As an example, Behles explained that

Figure 14. Screen shot of the *Stampede II* user interface.

From Behles, G. (1998). Reprinted with permission of the author.

"applying a voice-proportional modulation of 7 (semitones) to pitch detunes voice 1 by 7 semitones, voice 2 by -7 [negative seven] semitones, voices 3 by 14 semitones, voice 4 by -14 [negative fourteen] semitones, etc." (Behles, Starke, & Röbel, 1998, p. 49). The final modulation control allowed an associated parameter to react to changes in the intensity of the source audio. With positive values, this control caused the primary parameter to increase in value whenever the source sound grew louder and decrease as it became softer; negative values inverted this relationship. The three modulation elements were not mutually exclusive and could work together, enabling the composer to create complexity far beyond a uniform random distribution.

12. thOnk (van der Shoot, 1999)

thOnk was a popular application for granular processing that provided the composer with no controls other than selecting a source sound and a button to begin processing. The generation of parameter values was handled completely by random processes. The program therefore served as a way for the composer to totally automate exploration of granular processing. After rendering the output as a sound file, the composer could listen to the results and, if he or she so desired, use an audio editor to remove any unwanted portions. The web site that promoted the software hailed it as a cure for writers' block that allowed the composer to generate material "*without having to think at all*" (Audio Ease, 2002).

This type of interface was an extreme solution from software developers wishing to simplify the user experience and stood in stark contrast to the example provided by *Stampede II*. It places the composer in the role of editor, incapable of directly shaping the processing output. *thOnk*'s interface was the antithesis of the confrontational approach to interface design and illustrated Laske's (1988, 1989a, 1989b) theories; the programmer of *thOnk* has embedded compositional processes for determining parameter changes into the program, leaving the composer with no choice but to use them in producing sound output. *thOnk* was a clear demonstration of Lanksy's (1990) idea that using a program can be equivalent to "playing someone else's composition."

13. Csound granule generator (Lee, 2000)

Csound is one of the most popular audio synthesis languages currently in use by musicians around the world. Originally created by Barry Vercoe (1986/1997), the language is now "vast and ever expanding" (Boulanger, 2000, p. xxxviii). Its standard distribution package includes methods for producing both individual grains and granular events. Csound's granular event generator, known as *granule*, was written by Alan S. C. Lee (2000) and based on his prior work (Lee, 1995). Lee designed these events to be defined separately and sequentially within a score file. The individual entries would list the 22 parameters needed to generate each group of grains.

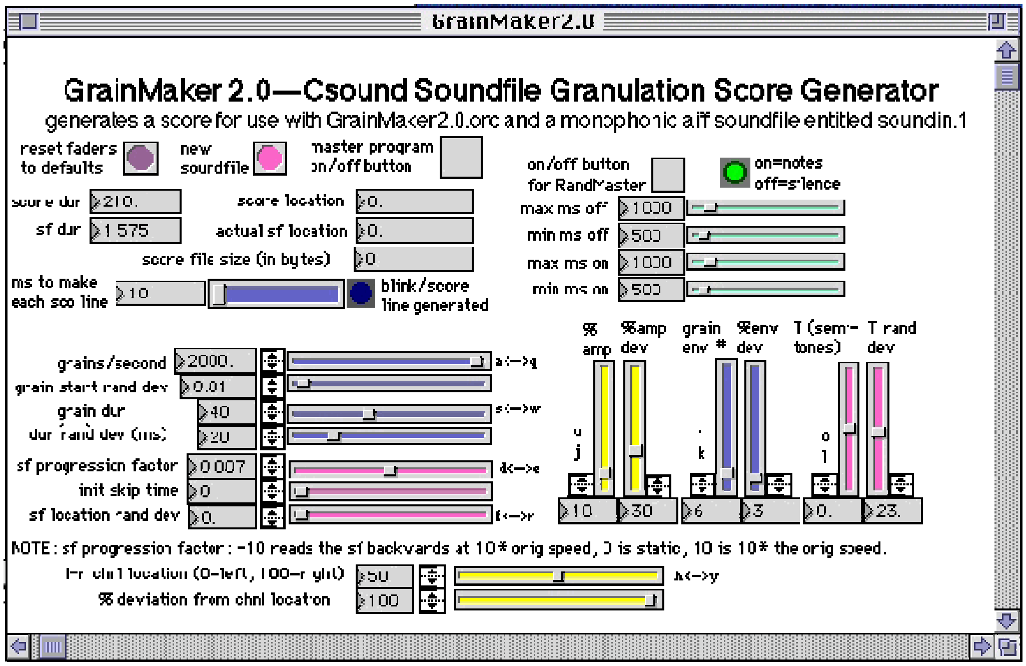
Lee included a unique parameter called "threshold" that provided "a simple design for skipping silent space within a sound sample" (Lee, 2000, p. 284). The composer could specify an amplitude value that the source sound must be above in order to be processed. If regions within the source were below this value, they were skipped as the sample offset progressed through the source. The motivation for including this parameter was prevent low-level audio, including background noise, from being processed. The implication was that processing such sound sources would produce uninteresting results.

The "size" parameter allowed composers to define the grain duration using seconds, a somewhat awkward choice given that durations are typically less than 50 ms. A secondary parameter controlled the amount of random deviation from the primary duration using a percentage. Grain delay was expressed in seconds by a "gap" parameter with a similar secondary percentage parameter defining random deviation. The attack and decay times were controlled separately through two parameters that express the percentage of the duration that each would occupy. The effect of Lee's use of percentages was analogous to the use of ratios in *Stampede II*; it caused parameters to be expressed indirectly through their relationships to one another.

14. GrainMaker 2 (Nelson, 2000a)

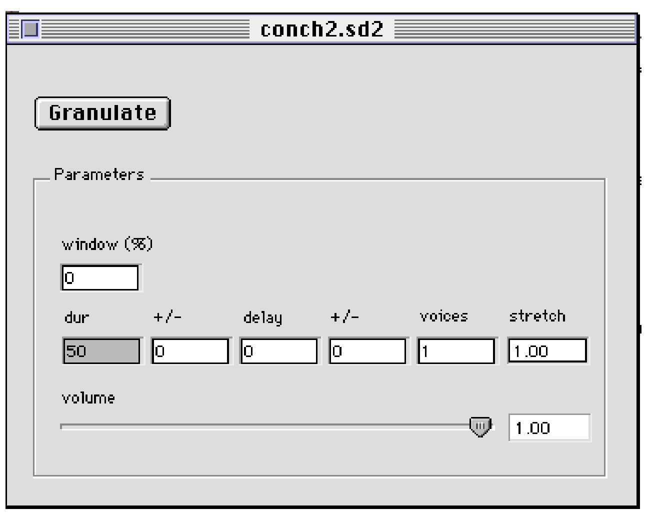
Another method for producing granular events with Csound involved using a score generating program similar to those already profiled for other languages

Figure 15. Screen shot of the *GrainMaker 2.0* user interface.



From Nelson, J. C. (2000a). Reprinted with permission of the author.

Figure 16. Screenshot of the *MacPod* user interface.



From Rolfe, C. and Keller, D. (2000). Reprinted with permission of the author.

(Helmuth, 1991; Roads, 1978, 1985). Jon Christopher Nelson created such a program called *GrainMaker 2* (Nelson, 2000a), which he developed prior to the inclusion of Lee's event generator. The program was actually developed in Max to produce the necessary score instructions for a secondary instrument program written in Csound. Nelson's interface (see Figure 15) provided slightly different controls than those for *granule*. His list of parameters included the following:

- base grain rate
- grain rate random deviation (added to base)
- base grain duration
- grain duration random deviation (added to base) (Nelson, 2000b, ¶ 8)

Using these parameters, his application produced a score file for his granular instrument program. Because random number generation was used, it could potentially produce different results when the same settings were used to generate multiple score files.

Compared to *granule*, the most important difference in *GrainMaker* was the ability to control grain voices according to the rate of grains per second (or grain frequency) instead of through grain duration and grain delay. The difference in control may have been easier for some composers to understand. Nelson's program also provided the user with a GUI, which may have been preferable for some composers to the text score entries required by *granule*.

15. MacPod (Rolfe & Keller, 2000)

MacPod was based on the programs of Truax (1988), but offered the advantage of requiring no special hardware. The programmers designed it as a stand-alone program that ran on an Apple Macintosh computer and it won an award at the 1999 Bourges Software Competition. Composers could use the program to produce granular processing textures with a scaled-down interface that offered only eight controls (see Figure 16), including some omissions from Truax's program. The program lacked any control over pitch shifting, forcing the output to retain the pitch of the source sound. The program was also incapable of real-time audio output, meaning a composer could not hear the effect of parameter changes immediately.

Despite any omissions, the program offered clear evidence of the direct influence earlier programs can have upon later developments. This was exemplified by the fact that *MacPod* controlled the rate of grain production via grain delay and random delay deviation. Although grain delay was part of Truax's original GSAMX program, its use within *MacPod* came after Truax (1994) concluded that grains per second was a "more intuitive variable" (p. 40). Remaining true to the original in this matter appears to have been more important than interface refinements for the programmers.

16. Granular Toolkit (Wolek, 2001)

The *Granular Toolkit* was developed by the present author for the purpose of producing granular processing effects within the Max/MSP environment (Cycling74, 2001), a commercially available update of the Max program used for some of the granular programs already described (Eckel, Rocha-Iturbide, & Becker, 1995; Lippe, 1994; Todoroff, 1995). The *Granular Toolkit* was not designed to be a single program. Instead it was conceived as a collection of objects for producing individual grains and patches for producing different types of granular events or granular voices. The sounds of these objects and patches were controlled in real-time and allowed the composer to hear the effect of changes immediately. The collection was organized using a naming system to aid in the easy identification of similar effects, because for most of the granular effects, two separate patches were created: one capable of loading a sound file for granular processing and a second used for processing a live audio source. A consistent user interface design was used across the different patches and objects.

The most complicated patch in the collection was designed to produce "cloud" textures like those described by Roads (1991) and provided the composer with many parameters (see Figure 17). The composer controlled the rate of grain production using a grain frequency parameter, with a secondary bandwidth parameter that randomized the grain periods. This randomization was

Figure 17. Screenshot of the *gran.cloud.live~* user interface.

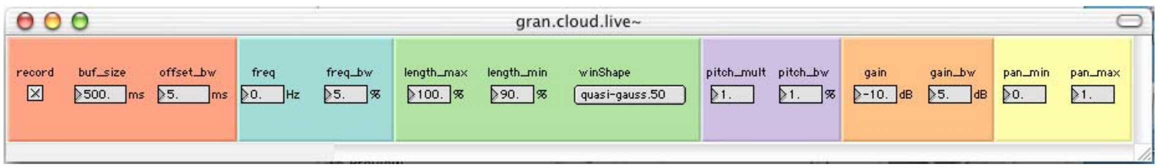
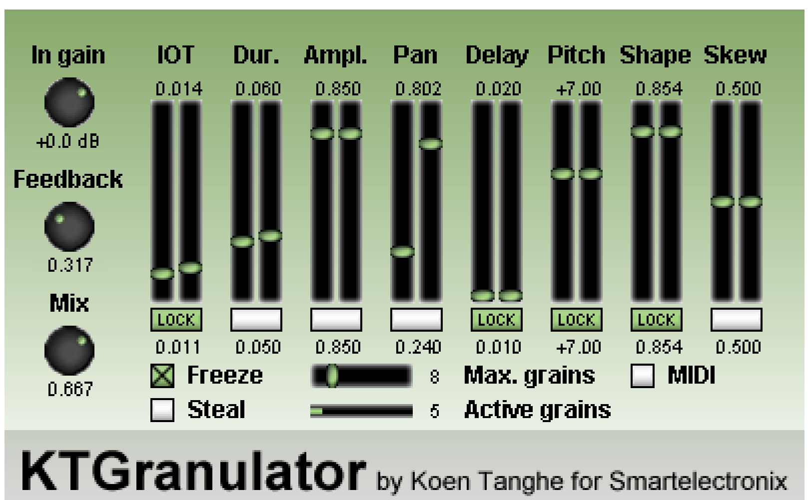


Figure 18. Screenshot of the *KTGranulator* user interface.



From Tanghe, K. (2003). Reprinted with permission of the author.

expressed as a percentage of the primary (or mean) frequency chosen. The grain duration, labeled "grain length", varied randomly between minimum and maximum length parameters, both expressed as a percentage of the grain period. This made duration dependant on the frequency parameter, so that the actual values used by the underlying algorithm will increase and decrease with corresponding changes in frequency. This is a situation similar to those already highlighted within other programs (Behles, Starke, & Röbel, 1998; Lee, 2000) and was a potential source of confusion for the composer.

17. KTGranulator (Tanghe, 2003)

Koen Tanghe designed *KTGranulator* as a VST (Steinberg Media Technologies GmbH, 1999) and AudioUnits (Apple Computer Inc., 2005) plug-in for performing granular processing inside a digital audio workstation environment. It was based on the granular processing implementation found in another application called AudioMulch (Bencina, 1998, 2001), which itself was inspired by GSAMX (Truax, 1988). Tanghe included many of the same parameters seen in the other programs, but provided very simple interface controls over randomization (see Figure 18). Two sliders were used to specify the minimum and maximum values for each parameter, allowing the composer to constrain the use of randomization in a consistent manner across all parameters.

A few of the parameter names were unique to this software. Tanghe gave *KTGranulator* a parameter called "delay", but this was not a control over grain delay. It controlled a simple delay buffer for the sound source input, similar to the pre-delay found on some reverb effect processors. "Inter-onset time" was instead used to control the succession of grains, a parameter equivalent to grain period. These parameters, as well as grain duration, used seconds as a unit of measurement. The use of seconds, rather than milliseconds, made it similar to the Csound *granule* generator (Lee, 2000).

Tanghe provided control over the maximum number of grain voices allowed and a display for the composer to monitor the number of voices being used at any given time. This was a useful interface element that allowed users to monitor the difference between their setting and the actual allocation of resources. The interface had 20 parameters in total, but the visual groupings of maximum and minimum sliders next to each other helped the interface seem less complex than some of the others profiled.

B. Focus for the Current Study

Each program discussed in the previous section offered a unique set of control parameters to composers. Based on the trends observed within these programs, the discussions and experiments that follow will focus on three key parameters: grain duration, "voice organization" and randomization. These provide a rich range of issues in the perception of granular processing and in the design of interfaces for its control.

Among these many programs, grain duration was most typically given its own parameter within a program's interface. However, in a few programs the duration was set indirectly. Eckel, Rocha-Iturbide and Becker (1995) and Todoroff (1995) set the duration by specifying constituent attack, sustain and decay times. Behles, Starke and Röbel (1998) and Wolek (2001) used a ratio related to the grain period to specify the grain duration, similar to those used in pulsar synthesis, a related microsound technique. Todoroff (1995) recognized that duration may not be a linear percept and compensated for this by introducing non-linear mappings into his interface.

Voice organization was usually controlled by some variation on the concept of density, including terms such as "grain rate" and "frequency." Truax (1988) introduced the idea of grain delay as an alternative control method. Others adopted grain delay as their programs' control parameter over voice organization (Lee 2000; Rolfe & Keller 2000; Todoroff 1995). Tanghe (2003) used the grain period to control voice organization.

Randomization was managed by either describing deviations from the mean or specifying minimum and maximum values. Deviations from the mean were used more often than the minimum-maximum method. The later method was used exclusively by Nelson (2000) and Taghe (2003). Helmuth (1991) used a variation on this method, coupling the two values with midpoint and tightness parameters to produce non-uniform random distributions. Wolek (2001) was the only program profiled in which different methods were used to manage randomization for different parameters.

The differences between the profiled programs in the areas of grain duration, voice organization and randomization may seem subtle (see Table 1 for an overview), but touch on underlying problems. When designing his own software (Wolek, 2001), the present author felt that these parameters were the most in need of revision. The deviations from the status quo by other programmers are evidence that they too considered alternatives for these parameters. Different situations may have motivated their changes, such as the conventions of a given programming platform or the needs of a specific composer, but the result is the same: a change from interface conventions. The current study will further affect the design of the control interface using the findings of original perceptual experiments as a basis for change. This study may potentially reaffirm the appropriateness of the granular processing parameters used by the majority of programs. However, the empirical method provides the needed objectivity to evaluate the differences discussed in this chapter and the competing concepts they represent.

Table 1. Summary of Control Parameters Used by Profiled Programs.

	Grain duration		Voice organization			Randomization	
Citation	Direct	Indirect	Density	Delay	Period	Mean	Min/Max
Roads, 1978, 1985	√ [a]		√			√ [b]	
Truax, 1988	√			√		√ [c]	
Lent, 1989 [d]							
Hulmuth, 1991	√		√				√ [e]
Lippe, 1994 [f]	√		√				
Eckel, et.al., 1995		√	√				
Todoroff, 1995		√		√			
Roads & Alexander, 1997	√		√			√	
Behles, et.al., 1998		√	√			√ [g]	
van der Shoot, 1999 [h]							
Lee, 2000	√			√		√	
Nelson, 2000	√		√				√
Rolfe & Keller, 2000	√			√		√	
Wolek, 2001		√	√			√ [i]	√ [j]
Tanghe, 2003	√				√		√

[a] Roads (1978) used a fixed duration of 20 ms. [b] Neither duration nor density was randomized, but randomization of frequency was via mean-bandwidth descriptors. [c] Duration was randomized; delay was not. [d] Parameters determined through analysis and not specified by composer. [e] Additional midpoint and tightness parameters enabled non-uniform random distributions. [f] Lippe (1994) used randomization, but never stated which method. [g] Additional randomization features were provided. See text for description. [h] All parameters algorithmically defined. [i] Provided for grain length. [j] Provided for grain frequency.

CHAPTER FOUR: EXPERIMENTAL METHODS AND PROCEDURES

The lack of uniformity among computer applications used to produce granular events and reflection upon his own prior software development lead the author to conclude that a more rigorous study of granular processing sounds was required. Inspired by Wessel's (1979) use of original empirical research to inform program interface development, the current author also sought to employ the same data analysis known as multidimensional scaling (MDS; Kruskal, 1964a, 1964b; Shepard, 1962a, 1962b). In experiments designed to use MDS, stimuli are presented in pairs to participants, who must then provide a similarity rating for each pair. These responses are used by the MDS algorithm to develop what Shepard (1962a) called an "*analysis of proximities*" (p. 126). This analysis system plots one point for each stimulus within a geometric space made of an arbitrary number of dimensions. This geometric arrangement offers a graphical representation of the relationships between all stimuli. Two points located close to one another in one or more dimensions would represent a pair that was considered to be very similar, while two points with a larger distance between them would represent a pair that was considered less similar.

Wessel (1979) focused on additive synthesis, an electronic music technique in which the amplitude and frequency of individual oscillators are independently controlled by envelope functions to produce time-varying spectrums. All stimuli used in his experiment were derived from the analysis of recorded orchestral instruments and subsequent resynthesis using the additive synthesis technique. His overriding goal was not unlike that of the current study, albeit focused on a different technique. He stated, “Additive synthesis requires a considerable if not overwhelming amount of explicit information, and we shall explore ways to reduce this quantity of data without sacrificing richness in the sonic result” (p. 46). However, his method was not without flaws that comprise the external validity of his findings. The most important factor related to the generalization of the results of this study is that he was the only subject to “participate” in the experiment. His data collection program also allowed him to repeat the individual sounds in a pair without a prescribed playback ordering and take breaks of indeterminate duration whenever he desired.

MDS has proven to be an effective analytical method for other studies focused on the perception of musical timbres. Although a direct comparison between the findings of the current study and this literature may prove elusive, it is useful to examine their experimental procedures for potential models. Grey (1977) examined the relationships between orchestral instrument timbres produced by additive synthesis with greater rigor than Wessel (1979). By examining the relationships between acoustical measures and his MDS solutions, he identified three dimensions of timbre in his findings: spectral distribution, spectral synchronicity and inharmonic energy. The first of these relates to our perception of how bright a timbre is, while the second is connected to how a timbre evolves over the tone's duration. The third dimension describes the strength of components that are not harmonically related to the fundamental frequency. He cited MDS as capable of revealing not just the acoustic attributes that listeners use to determine similarity, but "other factors involved in judgment strategies" (p. 1270) that may be less quantifiable. Participants were allowed to hear each pair only once, a requirement that may not be problematic for the experienced musicians he recruited. Grey used a larger pool of 20 subjects, but allowed 15 of them to repeat the experiment to produce a total of 35 data sets. He does not explain how the data may have differed had he omitted these second trials nor does he clearly justify their inclusion.

Kendall and Carterette (1991) focused on pairs of orchestral instrument timbres sounding in unison and found that the acoustic attributes related to their MDS solutions were very similar to those identified by prior experiments using isolated timbres. In their procedure, both music majors and non-music majors were recruited to form subgroups of equal size. These subgroups determined *a priori* allowed the authors to examine the possible effects of musical training on participants' similarity ratings, although no significant differences were found. Kendall, Carterette and Hajda (1999) used MDS in combination with other empirical methods to examine the similarity of synthesized timbres found on commercial synthesizers to their natural equivalents. Their findings revealed that spectral dissimilarities were the primary cause of any “unnaturalness” attributed to the synthesized timbres. Their four parallel experiments provided multiple perspectives on the topic being examined, but only one of them used MDS.

Iverson and Krumhansl (1993) ran a series of three experiments to examine differences between the onset and sustained portions of natural orchestral timbres. In addition to their original set of intact tones, they used digital sound editing to isolate these two segments from each other to produce two additional sets of stimuli. Separate groups of participants were recruited for each set of stimuli and asked to provide similarity ratings for all unique pairings within the assigned set. The MDS solutions produced for each experiment exhibited similar organizational trends, leading to the conclusion that the identifying characteristics found in timbre are present throughout an orchestral instrument tone. By segmenting their inquiry into a series of experiments with a consistent procedure, Iverson and Krumhansl developed a clear overview of their intended focus. The author felt this provided the most viable model upon which to base the experimental procedure for the current study and designed a similar series of three experiments using stimuli generated by granular processing. As will be shown in the following pages, the resulting MDS solutions were analyzed for evidence of organizational trends that could be used to inform the design of a new interface for granular processing.

A. Experiment Design and Preparation

1. Scope

Three experiments were designed to test how participants would relate examples of granular processing to one another. The author generated examples of granular processing for use as stimuli using a patch developed in Max/MSP (version 4.0.9 for Mac OS 9). Each stimulus represented a unique, static setting of the granular processing parameters. A limited number of parameters were selected for manipulation because of the exploratory nature of this study.

The effects of changes in grain length (also known as grain duration) and grain period were explored within these experiments. For both parameters, the Max/MSP patch allowed values to be entered using milliseconds as the unit of measurement. Randomization was applied to these parameters using a uniform-distribution random number generator that is part of the Max/MSP environment. Parameters for controlling the amount of randomization were expressed as bandwidths surrounding a mean value using a percentage of the mean (e.g., if grain length was set to 20 ms and randomized by 20%, then length values would fluctuate between 18 – 22 ms). This common descriptor was used when applying randomization to both the grain length and grain period parameters.

The first experiment used stimuli with differences in both grain length and grain period, without randomization applied to either. For the second experiment, differences were confined to the grain length with specific amounts of randomization applied to this parameter. For the third experiment, differences were confined to the grain period with specific amounts of randomization applied to this parameter. All other processing parameters were held constant within each experiment.

Because granular processing depends on an existing sound source, the choice of the sound source used to produce stimuli could affect participants' responses. In order to minimize this possibility, different sources were processed by identical program settings so that the sound source would constitute the only difference between certain stimuli pairs presented to the participants. Including sound source as an independent variable allowed the author to test the possible effects it had upon responses to specific program settings.

The stimuli used for these experiments are admittedly basic examples of granular processing. However, without first exploring how listeners perceive differences in such simple examples, it would be impossible to study more complex granular clouds. It is the author's hope the results of these experiments will provide the necessary basis for this line of inquiry to be continued in future experiments.

2. Research Questions

a. Primary

Processing differences between the stimuli should result in corresponding differences in the participants' similarity ratings. If participants were unable to detect differences among the stimuli, analysis of the MDS solution would reveal no clear organization. If their responses varied because of the processing differences, these variations should correlate to the program settings in some fashion. In addition, the review of literature highlighted processing descriptors that differed from those program settings used to generate stimuli. Testing for correlation between these alternative descriptors and the MDS solution could aid in identifying which of these provides the best description of the most salient features for participants.

Multiple stimuli were produced for each program setting using distinct sound sources. This allowed the experiments to test for the impact of sound source upon the participants' responses to similar program settings. If these differences did not impact responses, analysis of the MDS solution should reveal no evidence.

b. Secondary

Experience with electroacoustic music may have caused individuals to respond differently to stimuli within these experiments. That experience could have taken two different forms: listening or composing. Listening to electroacoustic music would increase the likelihood that a participant had previously heard granular sounds. Composing electroacoustic music would have provided participants with the opportunity to use them in a composition. By asking the participants about their prior experience with electroacoustic listening and composing, the data can be used to analyze potential differences in their ratings.

In order to avoid confusion resulting from conflicting personal definitions of electroacoustic music, participants were presented with the following definition at the session's beginning:

> A type of music in which sounds are created and/or manipulated using computers and/or electronic musical devices. This IS NOT restricted to musical works of a certain aesthetic or style. It is based solely on the methods of sound production. Composed sounds within such a piece must be reproduced over loudspeakers or headphones in order to be heard. If such sounds are paired with traditional acoustic instruments, the piece is still considered to be electro-acoustic.

Participants were then asked the following questions:

1. On average, how many electroacoustic works do you listen to per month (please round up to the nearest whole number)?
2. How many electroacoustic pieces have you composed in your lifetime (please estimate if necessary)?

The author developed operational definitions for these experiments that dictated a participant would be labeled as a "listener" or "composer" when the relevant answer was five or more. Responses provided by members of the listener and composer subgroups were compared to those made by the remaining participants to test for significant differences. Such differences would suggest that exposure to electroacoustic music affected subjects' responses within the experiment. If differences were found, each subgroup's responses were kept separate when formulating MDS solutions. If no differences were found, all participants from that experiment were treated as members of a single group.

3. Pre-Qualification of Participants

The goals of this study dictated the kind of individuals who were asked to participate. Because the data were to be used to inform the design a computer program interface, it was reasonable to assert that participants should be potential users of this new interface. Roads (2001) described an ideal interface for granular techniques as one "in which a musician specifies the desired sonic result in a musically descriptive language" (p. 28). Identifying musicians as the primary group of potential users for this new granular processing program is logical. Audio professionals, such as those working in recording or acoustics, were targeted as a second group for whom this program would be of interest. People working in these fields use advanced sound processing programs and, while they may not possess formal musical training, their work does require a high degree of listening expertise. Therefore, both musicians and audio professionals were recruited as participants for the current study.

In order to provide some measure of a person's experience, potential participants were asked to provide separate assessments of their total years of musical experience and years of experience with audio or recording technologies. Only those with four or more total years of experience between these two areas were asked to participate in the study. Answers to these questions were also recorded when the participants reported for their scheduled experiment session for later analysis.

4. Audio Stimuli Preparation

The author selected three sound sources to use in producing stimuli for the experiments. All three were recorded during a single session at the Summit Studio at Northwestern University. A Shure SM81 microphone connected to a Joe Meek VCTwinQcs pre-amp with the compression settings inactive captured sound from the recording booth. Equalization settings were kept to a minimum and remained constant for all three sources. The audio signal was routed to a Digidesign 888 audio interface connected to an Apple Macintosh dual-processor G4 desktop computer. The digital recordings were made using ProTools (version 5.1.1 for Mac OS 9) at a sampling rate of 44,100 samples/sec, 16-bits per sample.

A Schulmerick hand bell pitched at A5 (~440 Hz) was the first sound recorded so that it could be used as a reference for tuning the subsequent sound sources. After listening to the bell recording, a performer sounded an identical pitch on her flute, matching its intonation. She was instructed to maintain a straight, dark tone while playing. The same performer then sang the pitch on a "long e" vowel sound without vibrato. Several tones were recorded for each of the sound sources and the author chose one with the best quality from each set. These best tones were then bounced from the ProTools session into individual sound files and normalized. The three resulting sound files were later processed to produce stimuli for the experiments.

5. Preliminary Study

The total number of stimuli for each experiment would equal the product of the number of unique program settings and the number of sound sources. Participants must then provide a rating for each unique pairing. If the number of stimuli equals N, then the total number of unique pairings would equal [0.5 • N • (N+1)]. In order to keep the total number of pairings from exceeding a reasonable figure, the author had to limit either the number of settings or sound sources. After recording the sound sources, the decision was made that three was too many for the current study and would have required the number of unique program settings be reduced below a desirable level. In addition, too many timbres may have complicated interpretation of the experiments' results.

Timbre is itself regarded as a multi-dimensional percept, a concept known as timbre space (see Risset & Wessel, 1999, section XVI). Theoretically, three stimuli that differ only because of the sound sources' timbre could result in their differences occupying multiple dimensions of this study's MDS solutions. If this were to occur, the analysis could no longer focus on differences caused by granular processing. By reducing the number of sound sources to two, distinguishing between them should be easier for participants and should manifest itself in the results as a single-dimensioned dichotomy.

Because three sound sources had already been recorded, a preliminary experiment was conducted to identify the two that were the most dissimilar. Using a single group of processing settings (grain period = 93 ms; grain length = 43 ms), all three sound sources were processed to produce three preliminary stimuli similar to those used in the main experiments. Ten volunteers were informally asked to listen to the six possible pairs created among these three stimuli over headphones. The volunteers responded to each pair using an onscreen scrollbar, representing an unlabeled scale from 0 to 500. The responses for each pair were averaged together to produce a triangular matrix (see Table 2).

The most dissimilar pair was deemed the most desirable for stimuli creation in the main experiments. The author reached this conclusion because the motivation for including different sound sources was to determine whether the

Table 2. Mean Dissimilarity Ratings Between Three Source Sounds Processed Using Consistent Program Settings

Sound Source	1	2	3
1. flute	27.3		
2. vox_e	357.3	29.5	
3. bell	330.9	405.9	53.2

Note. Scale is from 0 – 500. N = 10. Program settings used were a grain period of 93 ms and a grain length of 43 ms.

Table 3. Program Settings Used to Generate Stimuli for Experiment 1

grain period (ms)	grain length (ms)
57	15
57	22
57	29
75	22
75	29
75	36
93	29
93	36
93	43

granular processing settings would interfere with normal timbre distinctions. By starting with the pair that has the least similarity, any confusion by subjects in subsequent experiments could be confidently attributed to such interference by the processing. The results from the preliminary study showed that participants rated the pairing of female vocal and hand bell as the most dissimilar. For this reason, the pairing was used to produce all of the stimuli for the three experiments in this study.

B. Experiment 1

1. Method

a. Stimulus Materials

A total of eighteen stimuli were produced for Experiment 1, resulting from the two sound sources selected in the preliminary study processed by a set of nine unique program settings. In addition, a sample of pink noise was processed using the same program settings to produce nine additional stimuli for use in the practice segment of the experiment. Grains were sampled from an offset of 200 ms without any randomization of this position. A pitch multiplier of 1.0 was used without any randomization of the multiplier, resulting in no pitch changes from the sound source. A Gaussian-shaped amplitude envelope (see Figure 7 from chapter 2) was applied to each sample segment. Specific pairings of grain length and grain period settings were the only parameters varied in the creation of these stimuli (see Table 3). No randomization was applied to either of these parameters.

Processing was performed on an Apple iBook using a Max/MSP patch developed by the author. The stimuli were recorded at 44,100 samples/sec, 16-bits per sample as AIFF files. Each stimulus was 700 ms long with an amplitude envelope that included a 100 ms linear fade-in and a 100 ms linear fade-out. The author chose to use this duration for fades 78because it was longer than any of the grain period settings being used, meaning that they would affect at least two complete periods within each stimuli. The resulting sound files were normalized using SoundHack (version 0.891; Erbe, 2002) and converted from AIFF to WAVE format with Sound App (version 2.6.1; Franke, 2002). This conversion allowed them to meet the file format requirements of the software used to manage the experiment procedure.

b. Participants

No one was compensated with money or course credit for participating in any of the three experiments. Only those volunteers who met the pre-qualification standards already delineated were scheduled for an experiment session. For Experiment 1, participants were recruited from the student population of the Audio Arts and Acoustics department at Columbia College Chicago. A total of 20 undergraduates participated in this experiment during February and March 2003.

c. Apparatus

Audio playback was presented over a single Tannoy System 10 DMT II monitoring speaker with concentric drivers. The speaker was placed directly in front of the listening position. The left channel of a Crown SA 30-30 amplifier was used to power the speaker. A Dell Latitude laptop computer running Windows 2000 handled all stimuli playback, with its sound output connected directly to the amplifier. Participants also used this computer to enter responses after the presentation of each stimuli pair for later analysis. An optical mouse was connected to the computer for participants to use when entering their responses into the computer. Stimuli playback and response recording was managed by a software program known as MEDS (version 2002-B1; Kendall, 2002) developed by Roger Kendall at UCLA.

Participants sat in a room typically used as a recording booth, part of the facilities at Columbia's Audio Technology Center. A chair was placed approximately 50 in. (127 cm) from the front of the speaker. Each participant was instructed by the author to sit in the chair and face the speaker when listening to the stimuli. The center of the speaker cone was placed upon a raised surface, with the center of the speaker cone measured at 55.5 in. (140.97 cm) above the floor. The chair height was adjusted so that each participant's head was approximately level with the speaker cone. The computer was placed to the right of the chair on a surface measured at 34.5 in. (87.63 cm) above the floor.

Nothing was placed between the chair and the speaker, providing a clear path between the participant and the audio source.

d. Procedure

Each participant was scheduled for an individual session with the author to complete the experiment tasks. When scheduling, each was informed that the experiment would last approximately 40 minutes. Before proceeding with the experiment, each had to review and sign a consent form in accordance with the requirements of Northwestern's Institutional Review Board (IRB). This form provided them with basic explanations of the motivation for the study and the procedure to be used.

When participants arrived for their scheduled session, they were guided to the room described in the Apparatus section. The author provided each with an orientation about the computer's role in recording responses and demonstrated how they could adjust the gain of the amplifier to a comfortable listening level. Each was instructed to face the speaker while listening and provided with the opportunity to ask any remaining questions they may have about the experiment. When confident they were ready to proceed, the author started the MEDS software and vacated the room for the duration of the session.

As the software began running the experiment, participants read the following text on the screen:

> This experiment will require you to make judgments about the similarity of pairs of sounds and should last no more than 40 minutes. At this point you should have already filled out the necessary consent forms for participating in this study. As a reminder, if you decide at any time that you would like to halt your participation in this study, we will stop immediately and any data you have provided up to that point will be discarded. Furthermore, if you should have any questions at a future date about your participation in this study, you may use the provided contact information to seek answers. Do you understand? If so, click OK to continue. If not, please ask the researcher for clarification before proceeding.

Clicking the "OK" button on the screen provided an additional form of consent for participants.

Next, participants entered their responses to the pre-qualifying questions they had already been asked prior to scheduling a session. The instructions on the computer prompted them to round up to the nearest whole number. The author deemed this necessary to prevent any confusion among participants about whether to include the current academic year. After these questions, they were presented with the definition of electroacoustic music already discussed (see Section A-2-b of this chapter). Participants were then prompted to respond to separate questions about their prior experience with listening and composing electroacoustic music. Answers to all four of these questions were used in the analysis to examine the role of experience in participants formulating their responses.

After answering these questions, participants were presented with the following text:

> You are about to hear pairs of sounds. For each pair that is played, consider the question, 'How much do you feel the first sound needs to be changed in order to make it the same as the second sound?' Click OK to proceed with the listening.

Responses to each pair were entered by controlling a horizontal, on-screen scrollbar with the computer's mouse. Participants could place the handle at any point between two ends, with the left end labeled "none" and the right labeled "a lot." The slider was devoid of any markings indicating the scale used, hiding values from 0 to 500. Presenting the similarity question in this manner was adapted from Iverson and Krumhansl (1993). The similarity question was kept in the upper left corner of the screen for the duration of the experiment so that subjects could refer back to it as necessary. Participants were allowed to repeat the playback of any given pair when necessary before providing their responses. The next pair was presented only after an answer for the proceeding pair had been finalized.

Before listening to the complete set of stimuli pairs based on the recorded sound sources, participants heard a series of practice pairings. These pairings contained the nine practice stimuli based on the pink noise sound source. All 45 unique pairings were presented in a random order to provide participants with a clear demonstration of the range of granular processing differences they would encounter. Participants were informed that this section was for practice purposes only and that their responses in this section would not be recorded. They were also instructed to use this section to adjust the playback level.

At the conclusion of the practice section, the participants were prompted that the pairings being investigated were about to commence and that answers from that point forward were being recorded. The pairings used the 18 stimuli generated for this experiment and included identity parings (i.e., the same stimuli heard twice). Reversed presentations of each pairing were not presented because a pilot study showed that ordering did not significantly impact participants' ratings. Participants were presented with all 171 unique pairings in a random order. Responses from this segment were used to create the MDS solutions.

2. Results

a. Descriptive Statistics

Participants in Experiment 1 reported slightly less than ten years combined experience with music and audio technologies (M = 9.93, SD = 4.14).

They reported twice as many years of experience with music (*M* = 6.80, *SD* = 4.30) as they did with audio technology (*M* = 3.13, *SD* = 1.43). However, there was a wider range of musical experience than audio experience.

Response to the questions about experience with electroacoustic music revealed a wide range of answers. Participants reported listening to a mean of 45.65 pieces per month (*SD* = 77.18) and having composed an average of 16.05 pieces in their lifetime (*SD* = 55.82). However, the median responses provided much lower values for both listening (*Mdn* = 15) and composing (*Mdn* = 3). Based on these numbers, the author concluded that some participants may have inflated their responses due to their interpretation of the electroacoustic music definition provided.

b. MANOVA

Participants were assigned to the listener/non-listener and composer/non-composers subgroups based on the operational definitions developed a priori. A large imbalance between the size of these subgroups caused concern, therefore an additional grouping function was developed a posteriori for analysis (see Table 4). The grouping was based on the relationship of each participant's response to the median for each question, thereby dividing the group in half. Both grouping systems were used as independent variables for a MANOVA of participants' responses to all 171 stimuli pairings.

The ratings showed no significant difference between members of the listener, $F(1, 16) = 0.41$, $p = .86$, and composer, $F(1, 16) = 0.27$, $p = .93$, subgroups and their counterparts. Dividing participants based on the median response also showed no significance for both the listening, $F(1, 16) = 0.39$, $p = .87$, and composing questions, $F(1, 16) = 0.64$, $p = .77$. These tests support the conclusion that participants' prior experience with electroacoustic music had no significant effect upon the ratings given to the stimuli pairs. Because no significant differences between these subgroups could be found, all participants were treated as members of a single group for further analytical purposes. The mean ratings for every presented stimuli pair were combined to form a single data set for developing the MDS solution.

c. MDS

The ALSCAL algorithm for performing MDS within SPSS (Windows Version 10.5) was used to develop the solutions provided in this document. Both two- (2D) and three-dimensional (3D) solutions were generated and the coordinates recorded for later analysis (see Appendix A). Computing Kruskal's stress formula 1 for the 2D solution produced a very low value (stress = 0.13587) and 90.5% of the variance (R^2) in the responses was accounted for in this configuration. Stress for the 3D solution decreased comparatively (stress = 0.08832) and the variance accounted for increased by 3.9% ($R^2 = 0.9444$).

Table 4. MANOVA Groupings for Experiment 1

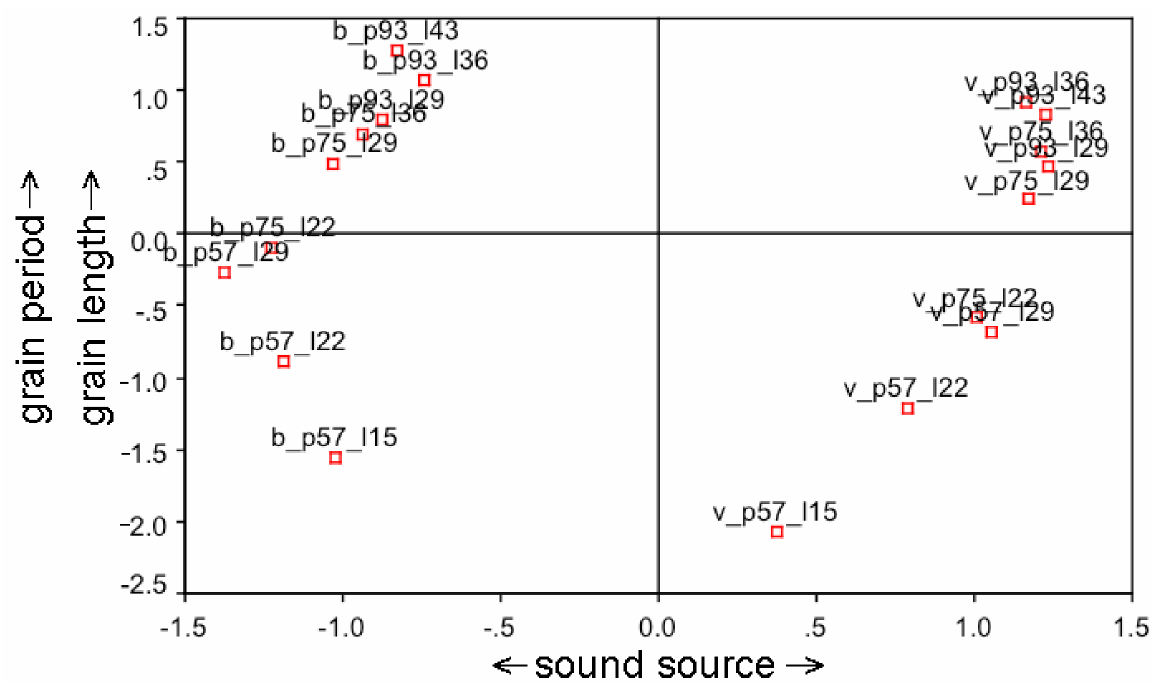
	Operational Definitions [a]		Median [b]	
Question responses	no	yes	<=	>
Listening (*n*)	3	17	12	8
Composing (*n*)	13	7	11	9

Note. *N* = 20. Groupings are based on participants responses to the pre-experiment questions outlined in the Secondary Research Questions section.

[a] Operational definitions were developed a priori and defined "yes" answers as all those answering 5 or more to the pre-experiment questions.

[b] For Listening: *Mdn* = 15. For Composing: *Mdn* = 3.

Figure 19. Plot of Two-Dimensional MDS Solution Coordinates from Experiment 1



Note. Labels upon the axes represent trends observed by the author in the placement of stimuli.

These values show that MDS solutions created a very good geometric fit for the participants' ratings.

A plot of the 2D solution (see Figure 19) contains several observable trends. Dimension 1, plotted along the x-axis, represents a difference in the source sound used to produce the stimuli. Those derived from the bell sound file are grouped on the left, while those derived from the voice sound file are grouped on the right. This appears to be clear evidence that subjects had no difficulty distinguishing between the two, even as the amount of processing applied to these sounds was varied. The trend is also clear in the raw coordinates for both the 2D and 3D solutions; within the first dimension of each, those stimuli derived from the bell source have negative values, while those derived from the voice source have positive values. Based on this trend, the author concluded that no confusion between source sounds was apparent within Experiment 1.

Dimension 2 of the 2D solution also presents a basic trend: Values for both grain period and grain length increase from bottom to top along the y-axis. However, it is difficult to grasp trends beyond this through visual analysis of the 2D plot. Visual analysis of the plotted 3D solution (see Figure 20) is also problematic because of the limitations of viewing such graphs within two dimensions. The author instead graphed each dimension separately to facilitate comparison between the two sound sources and their affect on the perception of the program settings. Figure 21 shows a clear separation between the two sound sources used within the first dimension. Figure 22 and 23 show common trends between the two sound sources within the second and third dimensions. These provided evidence that changes in the program settings were perceived as independent from the sound source.

The second dimension appears to be a continuous line, representing a corresponding change between the stimuli. At two points (between the third and fourth settings as well as the sixth and seventh settings) the line appears to flatten, indicating a possible equivalency between these stimuli pairs on this dimension. The third dimension has three distinct segments that coincide with the three grain periods used. The slopes of these segments have a common direction that corresponds with the increases in grain length at each grain period used. This single dimension offers the best evidence that participants did perceive nine unique program settings and attempted to relate them to one another.

Beyond these trends, it is not clear if the second and third dimensions represent a particular processing parameter. The inability to definitively label these graphical plots made it necessary to perform additional statistical analyses. Correlation of the MDS coordinates to program parameters was computed after results from all three experiments had been obtained. This process, which provided better evidence of the most salient features in these MDS solutions, will be described in the next chapter.

Figure 20. Plot of Three-Dimensional MDS Solution Coordinates from Experiment 1

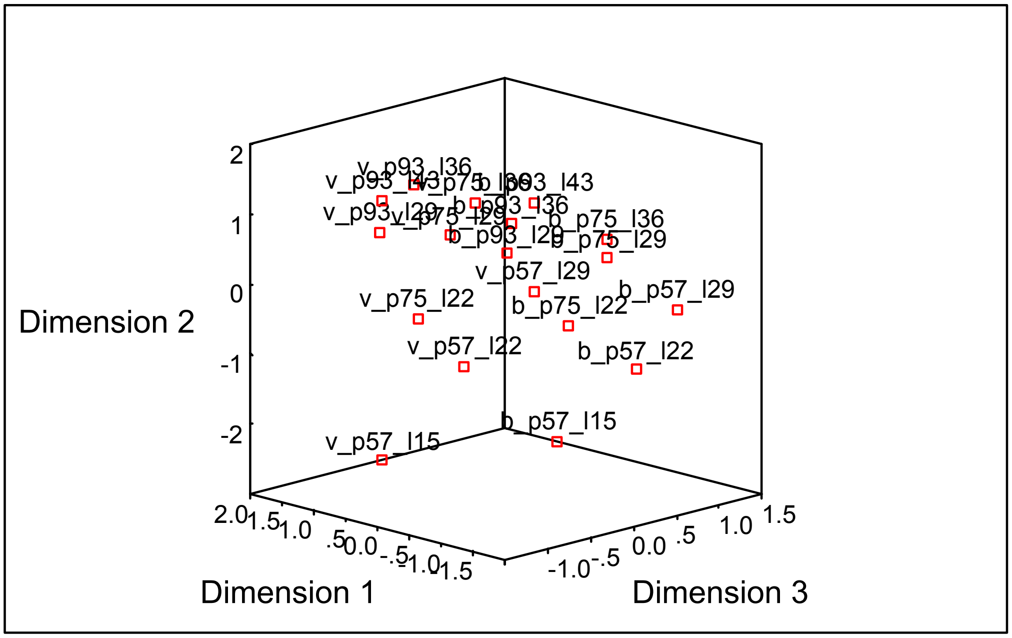


Figure 21. Dimension 1 Values from 3D MDS Coordinates for Experiment 1

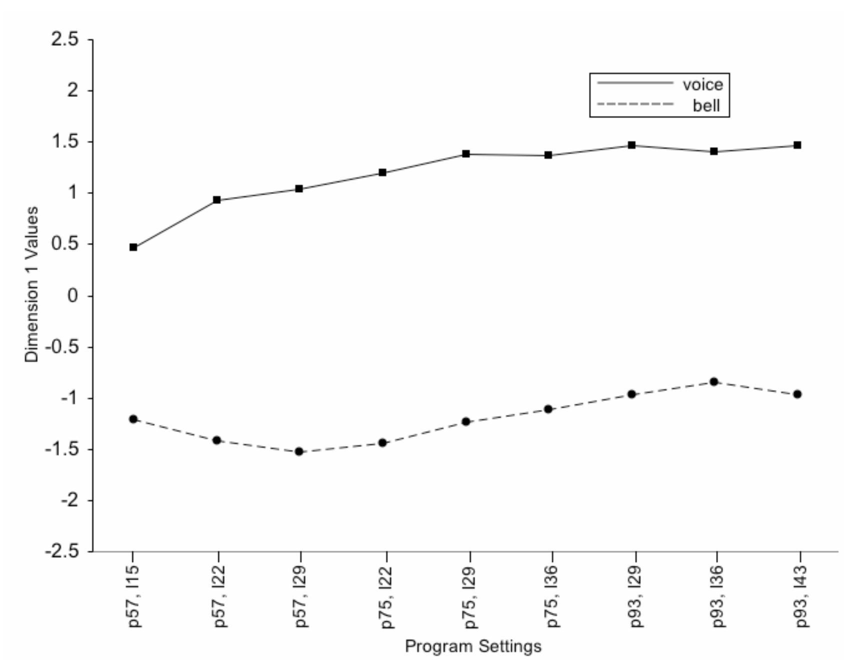


Figure 22. Dimension 2 Values from 3D MDS Coordinates for Experiment 1

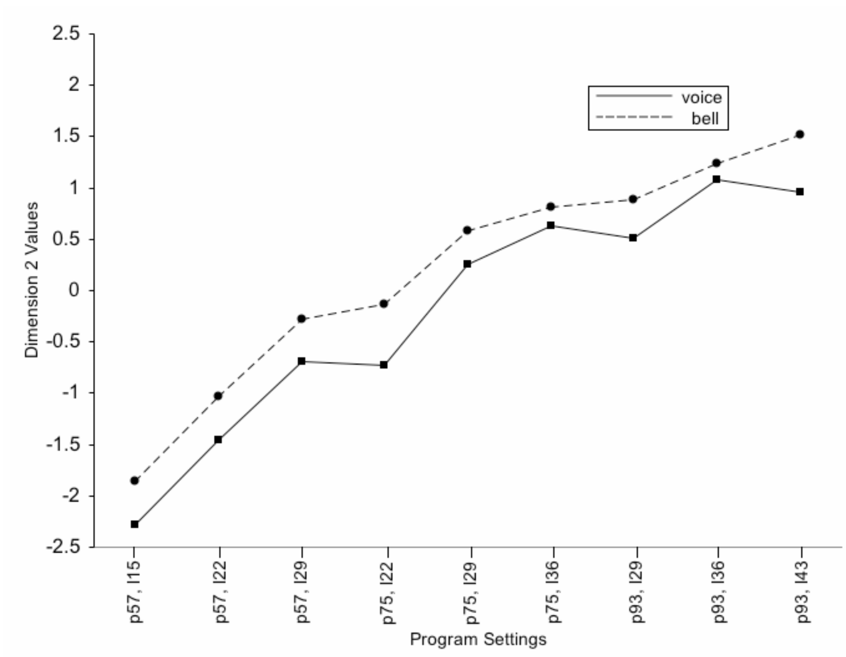
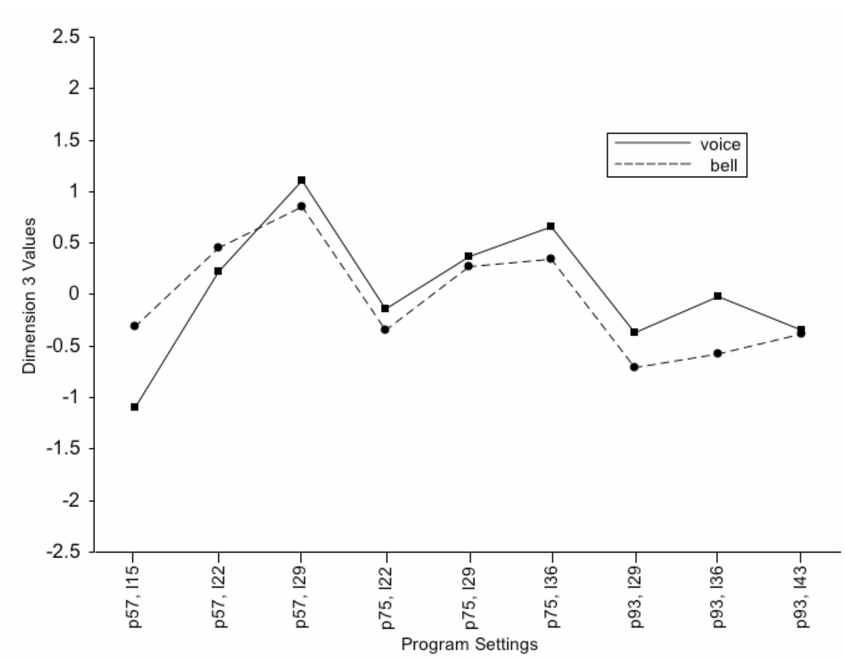


Figure 23. Dimension 3 Values from 3D MDS Coordinates for Experiment 1



C. Experiment 2

1. Method

a. Stimulus Materials

A new set of nine unique program settings were developed for Experiment 2. The two recorded sound sources and the pink noise sound source were processed using these settings to produce 18 actual and 9 practice stimuli, respectively. The grains were generated using a grain period of 75 ms and no randomization of this value. Specific pairings of grain length and length randomization were selected as the only parameters to vary among the stimuli (see Table 5). All other program settings were the same as Experiment 1.

The same Max/MSP patch was used to generate sound files. These files were prepared for use with MEDS using the same steps described for Experiment 1. A decision was made to make these stimuli slightly longer in duration than those used in Experiment 1 because the random fluctuations in length were less obvious at 700 ms. Therefore, each stimuli for Experiment 2 had a total duration of 1000 ms with fade-ins and fade-outs lasting 100 ms.

b. Participants

For Experiment 2, participants were recruited from the undergraduate and graduate student populations of the School of Music at Northwestern University. A total of 20 individuals participated in the second experiment during April 2003.

Table 5. Program Settings Used to Generate Stimuli for Experiment 2

grain length	
mean (ms)	bandwidth (%)
22	0
22	125
22	160
29	0
29	100
29	125
36	0
36	50
36	100

c. Apparatus

The same speaker from Experiment 1 was powered by the channel A (12 Ω rated output) of a Yamaha SR-50 surround sound amplifier. Audio signal was fed from the laptop to the Behringer Eurorack MX 1602 audio mixer, the output of which was connected to the amplifier. The remainder of the equipment used was identical to Experiment 1.

The room used for this experiment was located in Northwestern's Music Administration Building. Its designated use is for music cognition experiments conducted by faculty and students. The dimensions of this room were somewhat smaller than those of the room in Experiment 1, resulting in the speaker being placed in closer proximity to the listening position. Subjects were seated approximately 35 in. (88.9 cm) from the front of the speaker. The speaker was placed upon a raised surface with the center of the speaker cone measured at 47.5 in. (120.65 cm) above the floor. The chair used was incapable of height adjustments as in Experiment 1, but was of sufficient height to place participants' heads approximately level with the speaker cone. The laptop was placed to the left of the listening position on surface that was measured at 27 in. (68.58 cm) above the floor.

d. Procedure

The procedure was identical to that of Experiment 1.

2. Results

a. Descriptive Statistics

Participants in Experiment 2 reported more combined experience with music and audio technology (*M* = 15.45, *SD* = 5.51) than those from Experiment 1 (*M* = 9.93, *SD* = 4.14). This was due almost entirely to more musical experience (*M* = 12.25, *SD* = 3.94) than their counterparts from the first experiment (*M* = 6.80, *SD* = 4.30), something that was likely attributable to the fact that participants were recruited from a music school and included graduate students. The average audio experience (*M* = 3.20, *SD* = 4.27) was nearly identical to the first experiment (*M* = 3.13, *SD* = 1.43), albeit over a wider range.

Responses indicated that Experiment 2 participants had less prior contact with electroacoustic music than their counterparts from Experiment 1. They reported listening to a mean of 16.00 pieces per month (*SD* = 24.99 vs. *M* = 45.65, *SD* = 77.18) and having composed a mean of 13.50 pieces in their lifetime (*SD* = 23.38 vs. *M* = 16.05, *SD* = 55.82). Once again, the median responses differed greatly from the mean for listening (*Mdn* = 4) and composing (*Mdn* = 4).

b. MANOVA

Operational definitions were again used to divide participants into the subgroups used as independent variables for MANOVA of participants' ratings (see Table 6). Because the median response for both questions in this experiment was 4, the median subgroups were equivalent to the operationally defined subgroups. Therefore, it was unnecessary to use the median-defined subgroups as an a posteriori test.

Again, no significant difference was found between the listener and non-listener subgroups, $F(1, 16) = 0.45$, $p = .85$, as well as the composer and non-composer subgroups, $F(1, 16) = 0.60$, $p = .78$. This test allowed the author to again conclude that experience with electroacoustic music had no significant effect upon the ratings provided and consider all participants together as a single group. Ratings were again averaged into a single data set for use with the MDS algorithm.

c. MDS

The same procedure used in Experiment 1 generated 2D and 3D MDS solutions for Experiment 2 (see Appendix A). Stress values were again low for both the 2D (stress = 0.13045) and 3D (stress = 0.08268) solutions. The amount of variance accounted for in the 2D (R^2 = 0.92762) and 3D (R^2 = 0.96403) solutions was once again high. These values support the conclusion that the MDS solutions for Experiment 2 were a very good fit for the similarity ratings.

The 2D solution was graphed (see Figure 24) so that trends could again be observed. The x-axis, again representing Dimension 1, shows the same division between source sounds that was observed in Experiment 1. The raw coordinates support this observation; the first dimension of both the 2D and 3D solutions contain negative values for stimuli based on the voice sound source

Table 6. MANOVA Groupings for Experiment 2

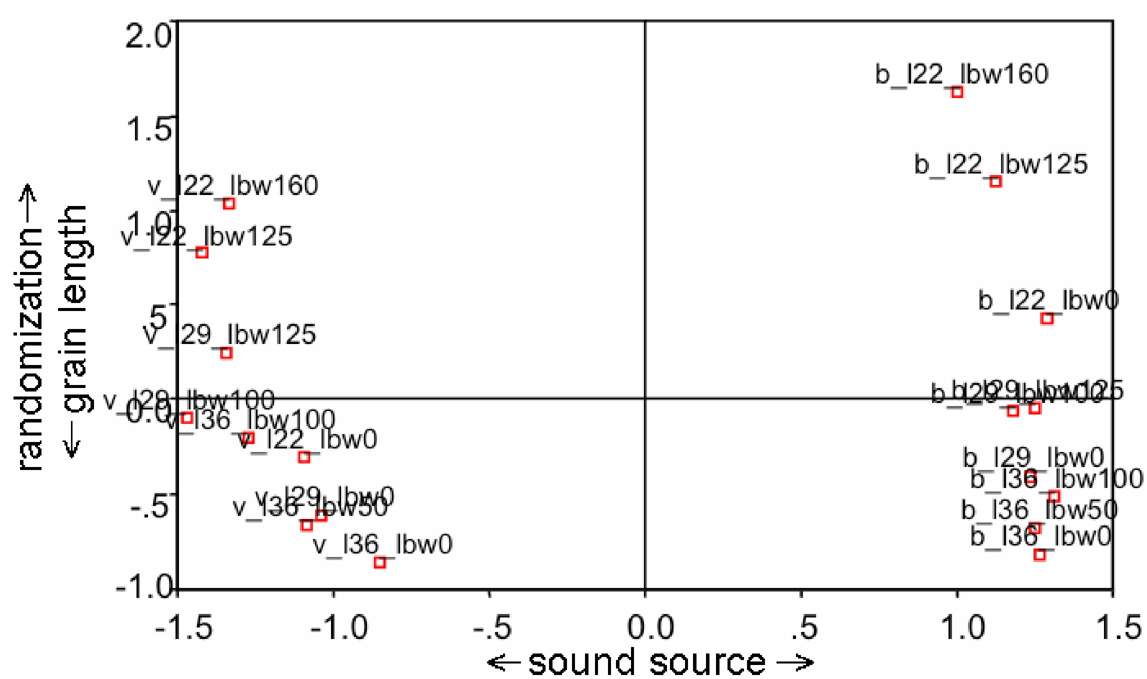
	Operational Definitions [a]		Median [b]	
Question responses	no	yes	<=	>
Listening (*n*)	12	8	12	8
Composing (*n*)	12	8	12	8

Note. *N* = 20. Groupings are based on participants responses to the pre-experiment questions outlined in the Secondary Research Questions section.

[a] Operational definitions were developed a priori and defined "yes" answers as all those answering 5 or more to the pre-experiment questions.

[b] For Listening: *Mdn* = 4. For Composing: *Mdn* = 4.

Figure 24. Plot of Two-Dimensional MDS Solution Coordinates from Experiment 2



Note. Labels upon the axes represent trends observed by the author in the placement of stimuli.

and positive values for those based on the bell sound source. The feature supports the conclusion that participants exhibit no confusion between the source sounds used.

Studying the y-axis, which represents Dimension 2 of the 2D solution, revealed two observable trends. First, the mean grain length values increase from the top to the bottom of the graph. Second, the length bandwidth values exhibit a contrary pattern, increasing from the bottom to the top of the graph. The pattern is at least partly attributable to the set of parameter values used for the processing, in which larger randomization values were paired with shorter grain length values (see Table 5).

The values from each of the dimensions of the 3D MDS solutions were graphed separately, just as they were for the first experiment results. Figure 25 shows a clear separation between the two sound sources used in the first dimension. Figure 26 and 27 show values from the second and third dimensions, respectively, that appear to modulate independently of the sound sources used for each stimulus. The second dimension shows a similar three-segment pattern to the one observed in the third dimension of the Experiment 1 results. This suggests that participants did perceive the three common grain lengths and the three different bandwidths used with each. The third dimension exhibits less coordination between the stimuli derived from different sound sources and common program settings. Given the prior evidence of Dimension 1, this is likely

Figure 25. Dimension 1 Values from 3D MDS Coordinates for Experiment 2

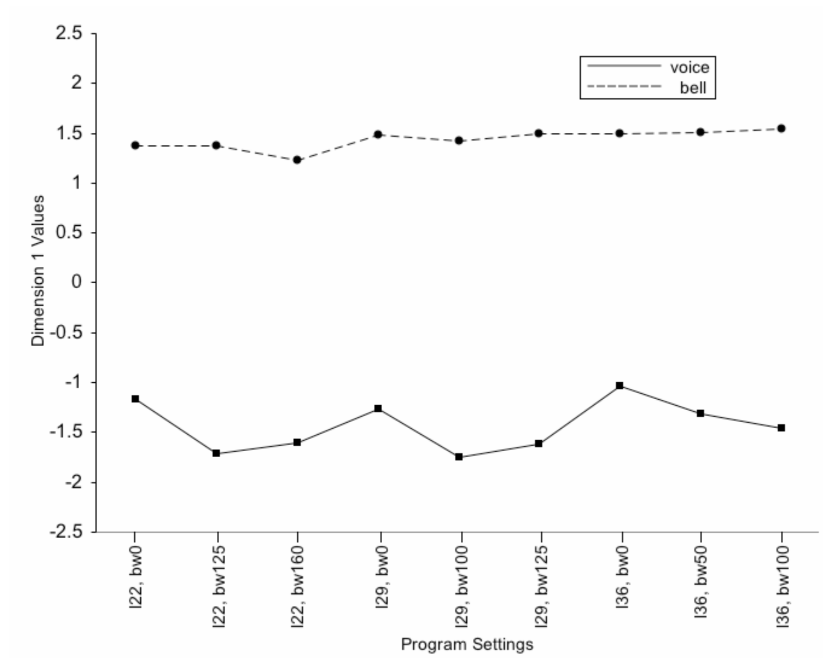


Figure 26. Dimension 2 Values from 3D MDS Coordinates for Experiment 2

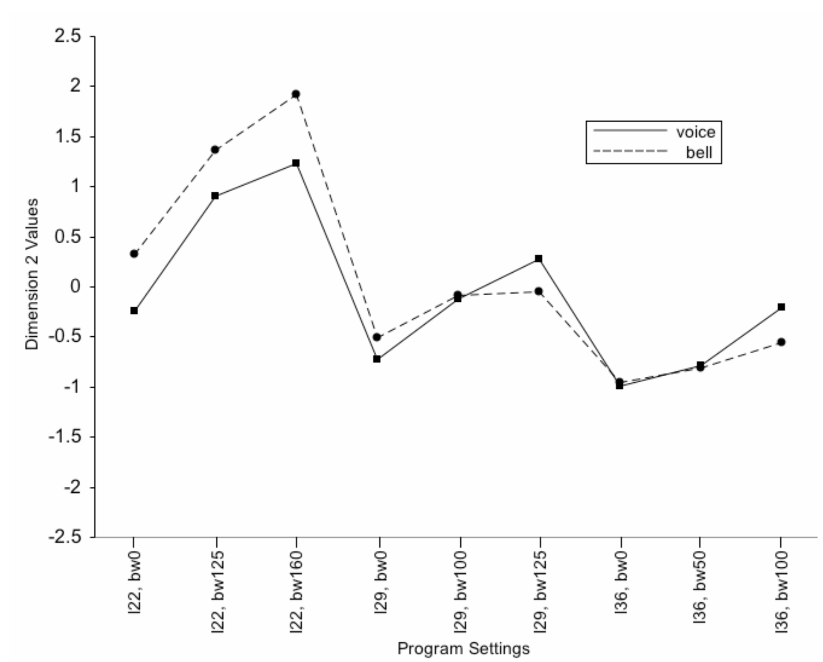
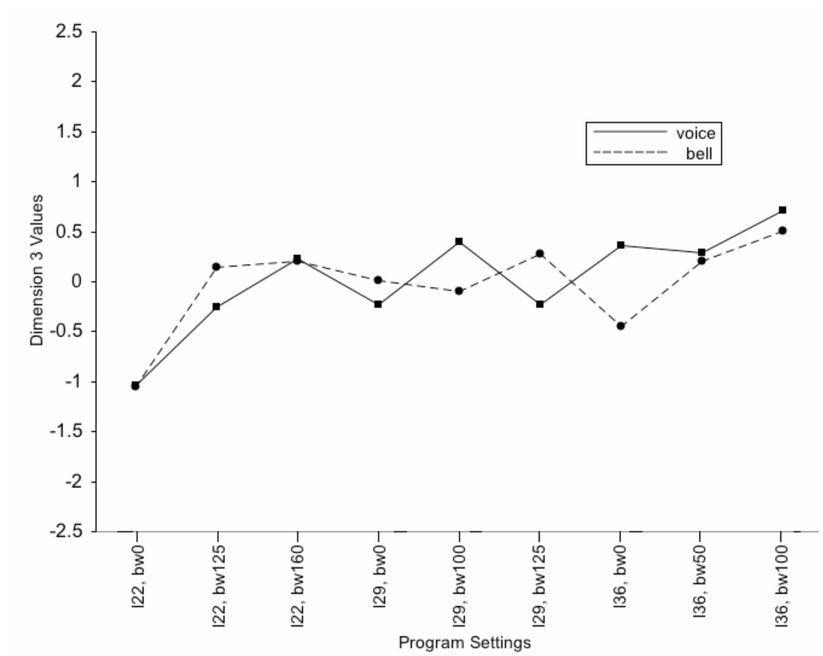


Figure 27. Dimension 3 Values from 3D MDS Coordinates for Experiment 2



not the result of confusion between the two sound sources and more plausibly attributable to the addition of randomization into the processing. Because of the randomization applied to parameters for individual grains, stimuli that are generated using the same settings may result in different features upon output. The analysis in the following chapter will further illuminate this point.

D. Experiment 3

1. Method

a. Stimulus Materials

A third set of nine unique program settings were developed to produce 18 actual and 9 practice stimuli for Experiment 3. The grains were generated with a grain length of 29 ms and no randomization of this value. Specific pairings of grain period and period randomization were selected as the only parameters to vary among stimuli (see Table 7). All other program settings were the same as in the first two experiments. The sound files were of the same duration as those used for Experiment 2 and prepared for use with MEDS using an identical method.

b. Participants

Participants were again recruited from the student population of the Audio Arts and Acoustics department at Columbia College Chicago. A total of 22

Table 7. Program Settings Used to Generate Stimuli for Experiment 3

grain period	
mean (ms)	bandwidth (%)
57	0
57	50
57	100
75	0
75	100
75	125
93	0
93	125
93	160

Table 8. MANOVA Groupings for Experiment 3

	Operational Definitions [a]		Median [b]	
Question responses	no	yes	<=	>
Listening (*n*)	5	17	11	11
Composing (*n*)	9	13	12	10

Note. N = 22. Groupings are based on participants responses to the pre-experiment questions outlined in the Secondary Research Questions section.

[a] Operational definitions were developed a priori and defined "yes" answers as all those answering 5 or more to the pre-experiment questions.

[b] For Listening: Mdn = 17.5. For Composing: Mdn = 5.

undergraduates participated in the third experiment during April and May 2003. None of the participants for Experiment 3 had taken part in Experiment 1.

c. Apparatus

The equipment and room were identical to those used in Experiment 1.

d. Procedure

The procedure was identical to that of Experiments 1 and 2.

2. Results

a. Descriptive Statistics

Participants in Experiment 3 reported a level of combined music and audio experience averaging 13.05 years (*SD* = 4.72). This level was between those from the first two experiments (Experiment 1: *M* = 9.93, *SD* = 4.14; Experiment 2: *M* = 15.45, *SD* = 5.51). Music experience alone averaged 8.73 years (*SD* = 4.70), a level also between those from the first two experiments (Experiment 1: *M* = 6.80, *SD* = 4.30; Experiment 2: *M* = 12.25, *SD* = 3.94). However, the participants' audio experience averaged 4.32 years (*SD* = 2.26), a figure that was higher than previous levels (Experiment 1: *M* = 3.13, *SD* = 1.43; Experiment 2: *M* = 3.20, *SD* = 4.27).

The mean level of electroacoustic listening (*M* = 33.73, *SD* = 55.65) was greater than Experiment 2 (*M* = 16.00, *SD* = 24.99), but did not reach the level reported in Experiment 1 (*M* = 45.65, *SD* = 77.18). Participants reported the lowest mean level of composition experience observed among the three experiments (*M* = 11.18, *SD* = 13.45; Experiment 1: *M* = 16.05, *SD* = 55.82; Experiment 2: *M* = 13.50, *SD* = 23.38). The medians once again differed greatly from the means for both listening (*Mdn* = 17.5) and composing (*Mdn* = 5).

b. MANOVA

As in Experiment 1, the a priori operational definitions for grouping resulted in unequal numbers between the listener and non-listener subgroups (see Table 8). The median was again used to divide the participants a posteriori in order to provide an additional independent variable for analysis. MANOVA was performed on the 171 ratings provided by each participant to determine if any significant difference existed between participants based on either the operational definitions or relationship to the median.

Unlike the first two experiments, there were significant differences found between the listeners and non-listeners, $F(1, 18) = 2125.30$, $p = .02$, as well as the composers and non-composers, $F(1, 18) = 2507.80$, $p = .02$. When divided according to the median, there were no significant differences for listening, $F(1, 18) = 120.49$, $p = .07$, or composing, $F(1, 18) = 7.79$, $p = .28$, subgroups. Because significance was found, the operationally defined subgroups were kept intact for the purpose of performing MDS analyses. The responses for each participant were grouped as part of the listener ($n = 5$) or non-listener ($n = 17$) subgroup, as well as the composer ($n = 9$) or non-composer ($n = 13$) subgroup.

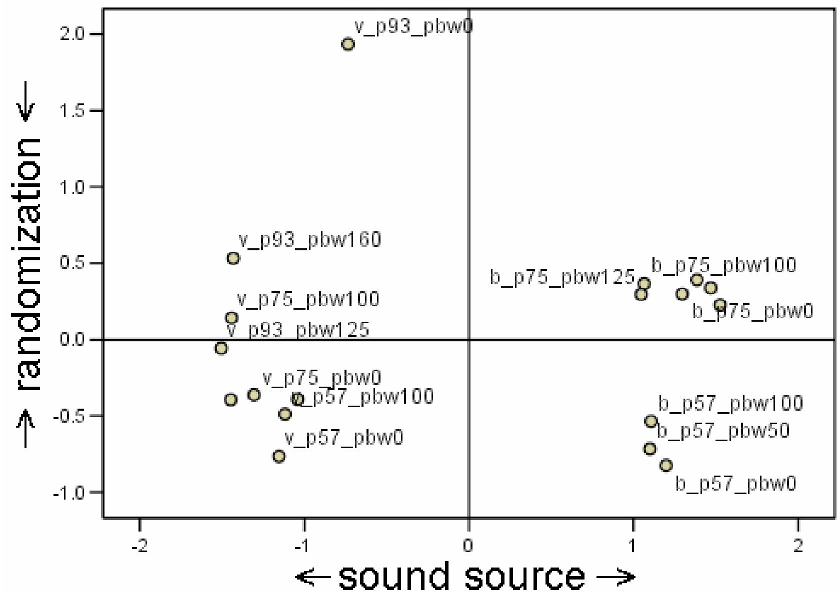
For each of the four subgroups, the 171 mean ratings formed a unique data set to be used with the MDS algorithm.

c. MDS

SPSS (Windows version 12.0) computed the ALSCAL algorithm to produce the MDS solutions described in this section. Coordinates for 2D and 3D solutions were recorded for the non-listener, listener, non-composer, and composer subgroups (see Appendix A). Stress values were similar to those observed in the first two experiments, with lowest reported for the 3D listeners solution (stress = 0.07412) and the highest reported for the 2D non-listeners solution (stress = 0.16865). The variances were also high, with all reporting a level at 89% or above. The lowest variance was reported for the 2D non-listeners solution (R^2 = 0.88988).

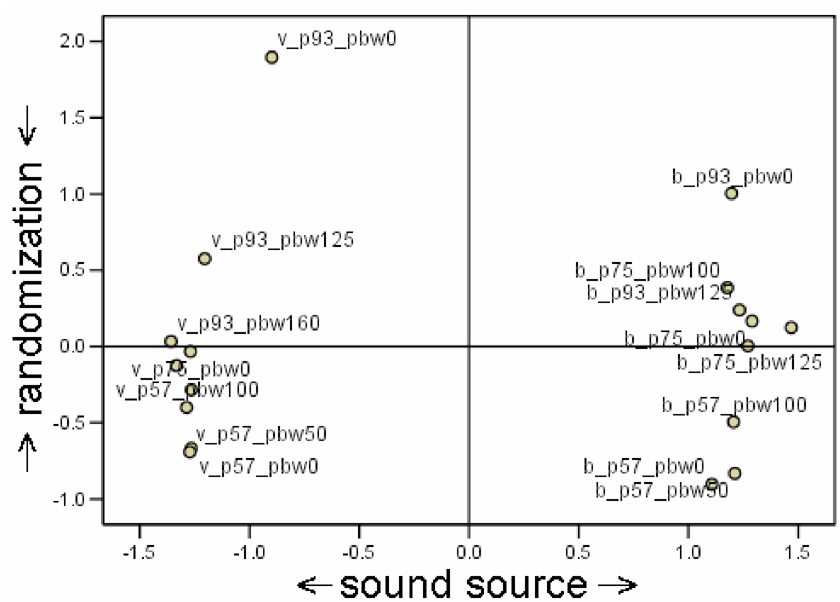
The 2D plots (see Figures 28 through 31) show the same separation between sound sources that was found in the results of the first two experiments. In most of the plots, the grain period increases in one direction across the second dimension. There are also a few misplaced stimuli that break with this trend, therefore confidence in identifying this dimension is not as high as it was with previous results. Only the composer subgroup (n = 13) appears to be without misplacements across this dimension. Another trend is the gathering of stimuli with higher randomization settings in the middle of the dimension. This is different than Experiment 2, where randomization appeared to increase in the

Figure 28. Plot of Two-Dimensional MDS Solution Coordinates from Experiment 3 (Non-listeners, n = 5)



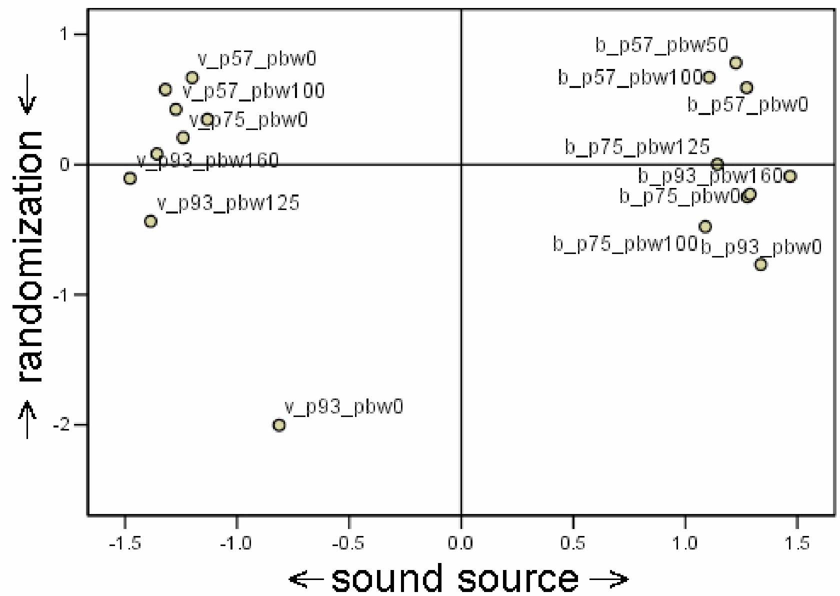
Note. Labels upon the axes represent trends observed by the author in the placement of stimuli.

Figure 29. Plot of Two-Dimensional MDS Solution Coordinates from Experiment 3 (Listeners, n = 17)



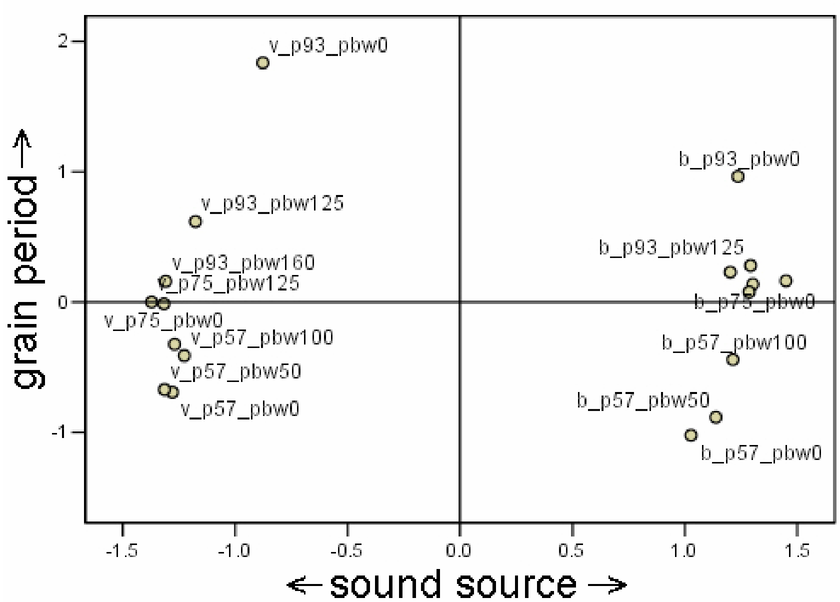
Note. Labels upon the axes represent trends observed by the author in the placement of stimuli.

Figure 30. Plot of Two-Dimensional MDS Solution Coordinates from Experiment 3 (Non-composers, n = 9)



Note. Labels upon the axes represent trends observed by the author in the placement of stimuli.

Figure 31. Plot of Two-Dimensional MDS Solution Coordinates from Experiment 3 (Composers, n = 13)



Note. Labels upon the axes represent trends observed by the author in the placement of stimuli.

Figure 32. Dimension 1 Values from 3D MDS Coordinates for Experiment 3 (Non-listeners, n = 5)

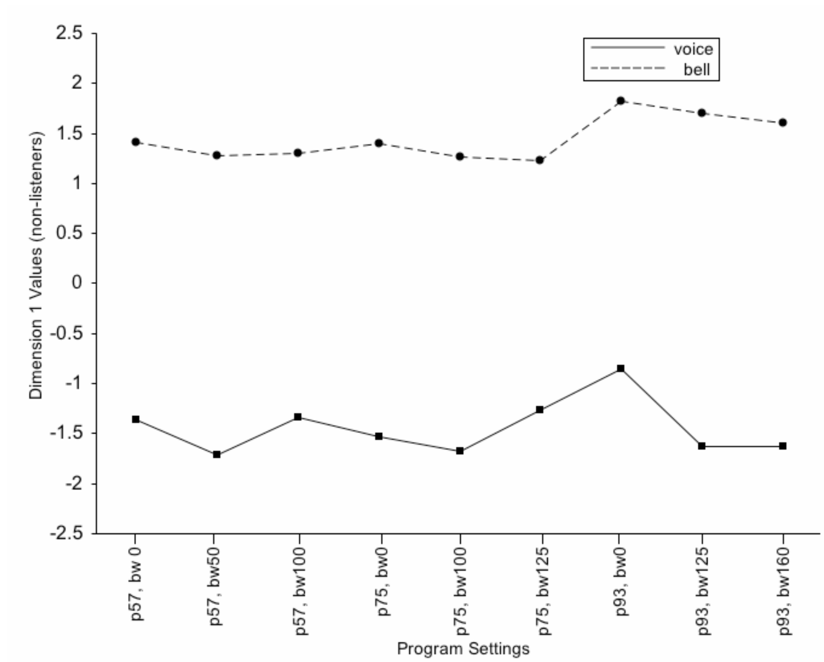


Figure 33. Dimension 1 Values from 3D MDS Coordinates for Experiment 3 (Listeners, n = 17)

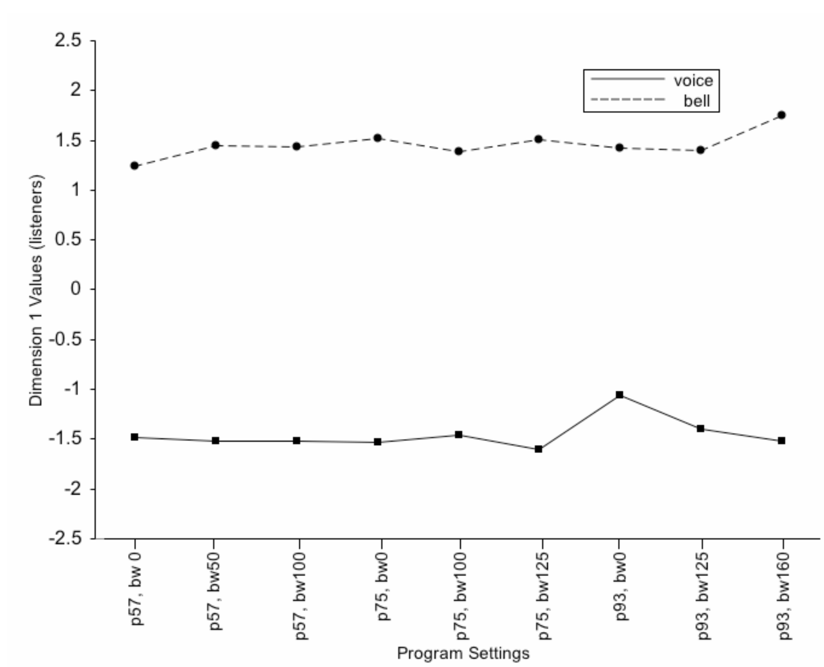


Figure 34. Dimension 1 Values from 3D MDS Coordinates for Experiment 3 (Non-composers, n = 9)

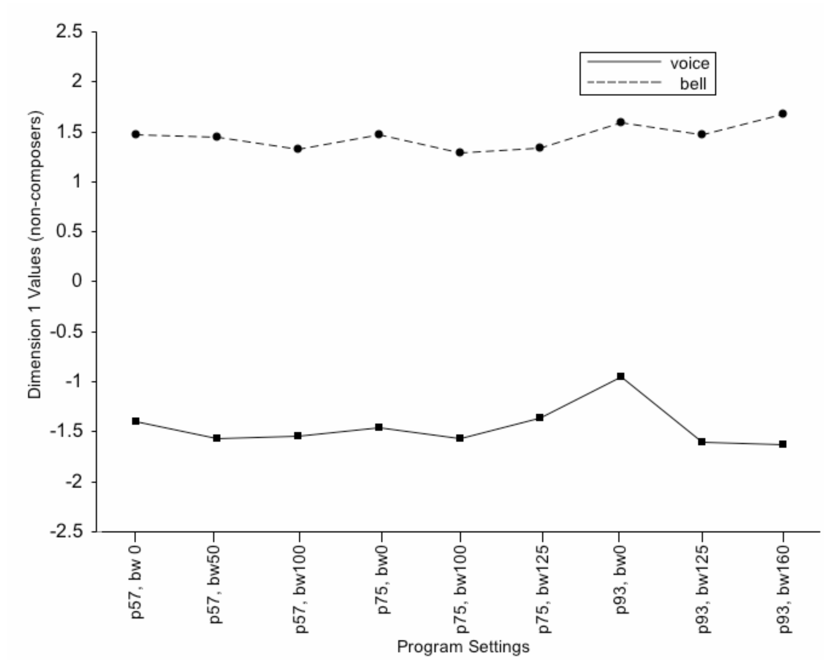


Figure 35. Dimension 1 Values from 3D MDS Coordinates for Experiment 3 (Composers, n = 13)

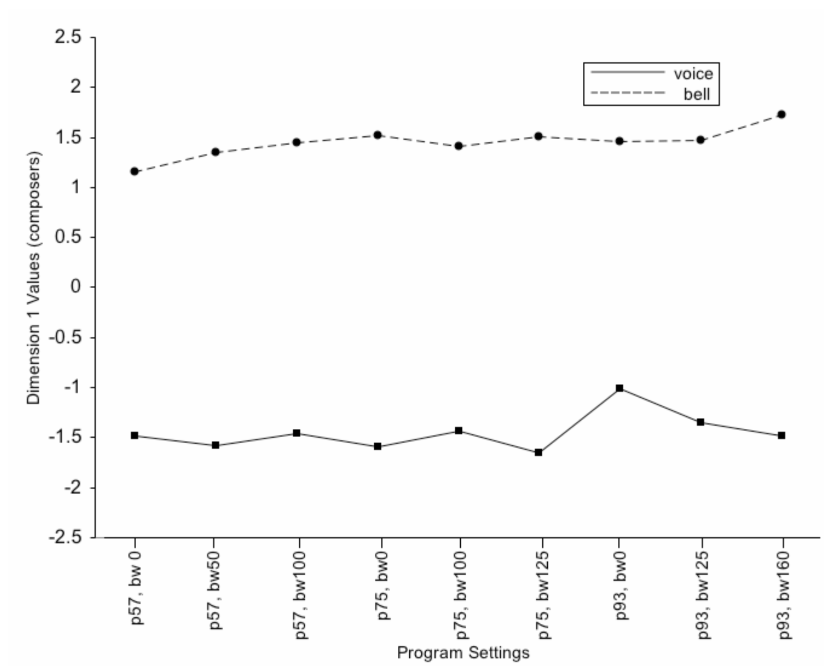


Figure 36. Dimension 2 Values from 3D MDS Coordinates for Experiment 3 (Non-listeners, n = 5)

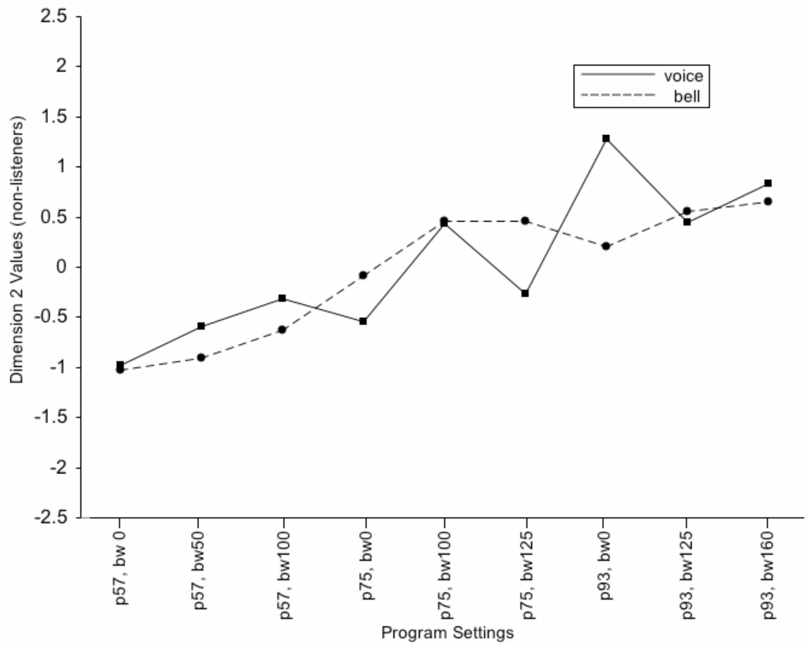


Figure 37. Dimension 2 Values from 3D MDS Coordinates for Experiment 3 (Listeners, n = 17)

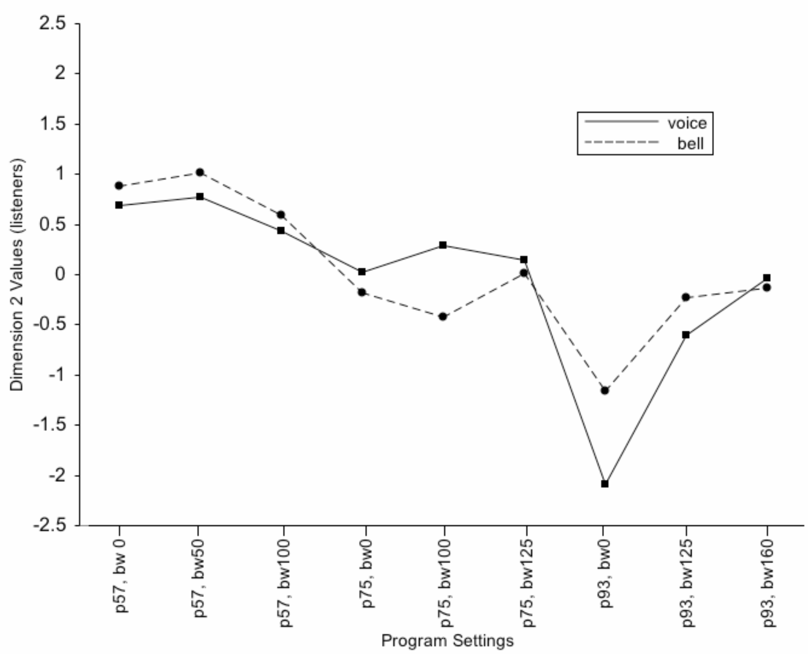


Figure 38. Dimension 2 Values from 3D MDS Coordinates for Experiment 3 (Non-composers, n = 9)

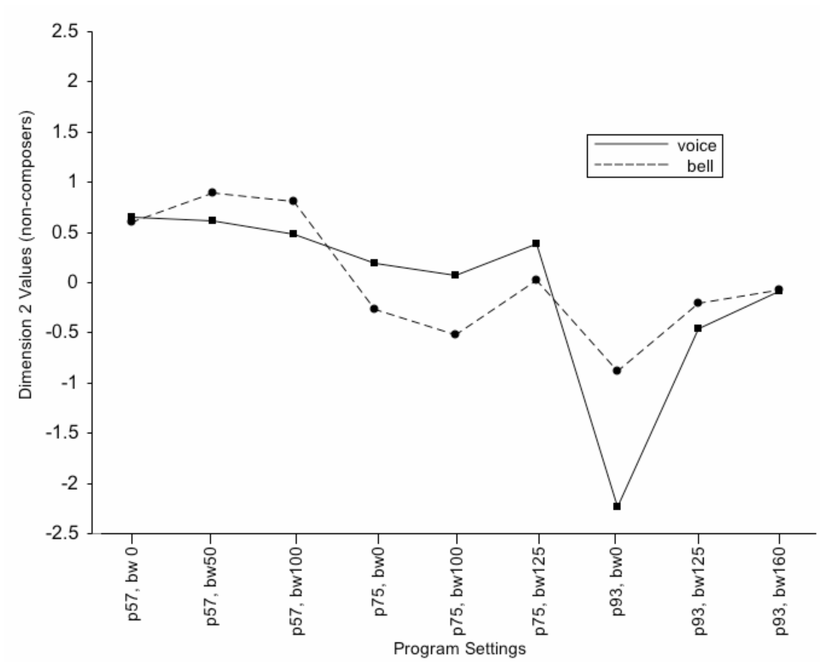


Figure 39. Dimension 2 Values from 3D MDS Coordinates for Experiment 3 (Composers, n = 13)

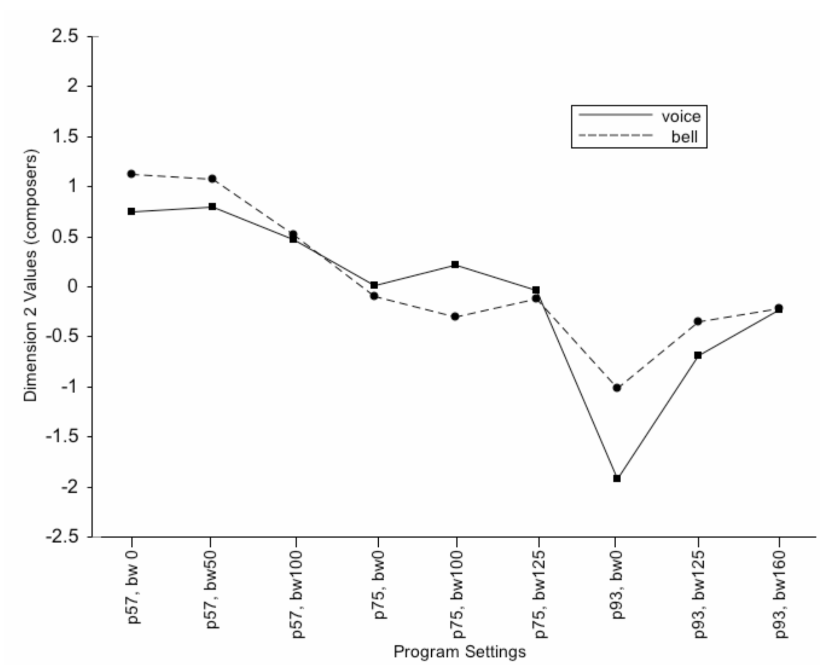


Figure 40. Dimension 3 Values from 3D MDS Coordinates for Experiment 3 (Non-listeners, n = 5)

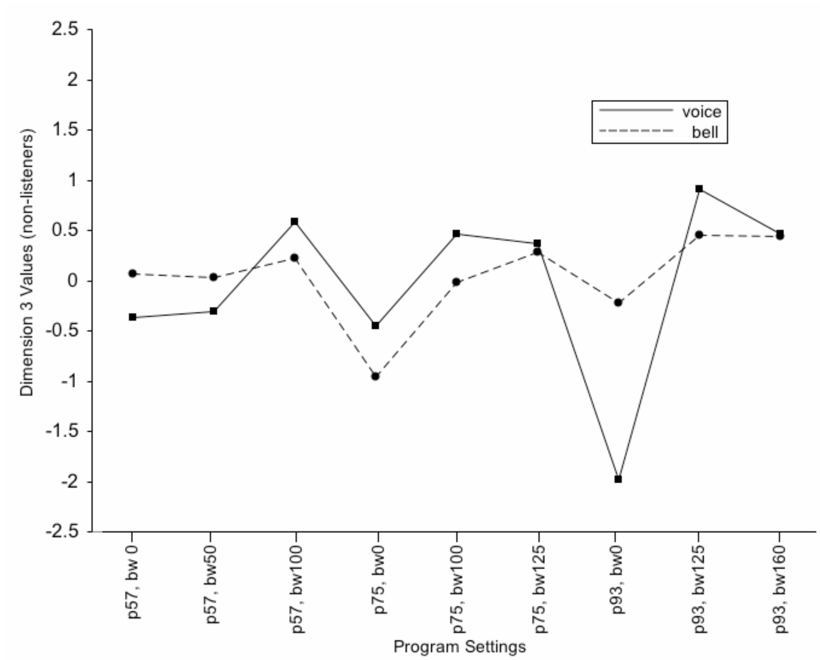


Figure 41. Dimension 3 Values from 3D MDS Coordinates for Experiment 3 (Listeners, n = 17)

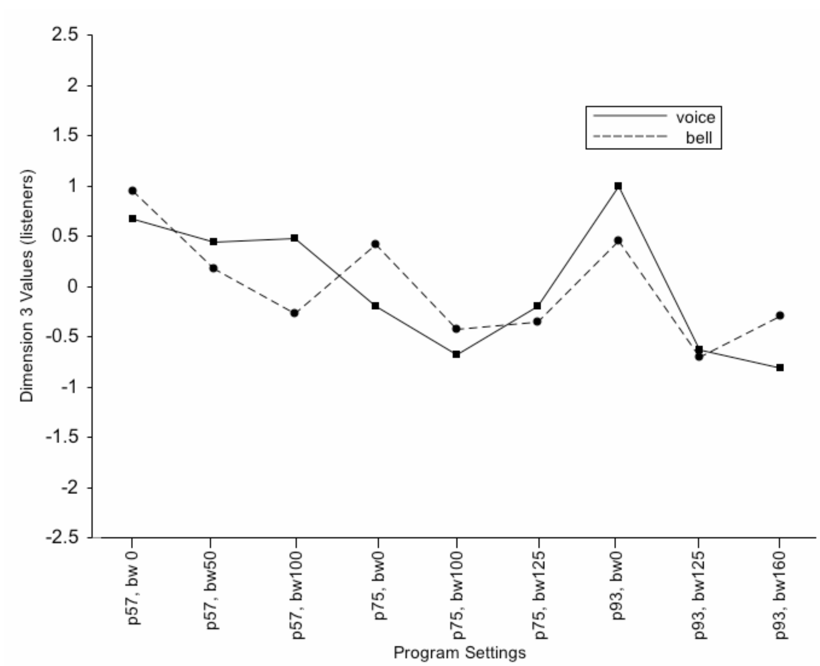


Figure 42. Dimension 3 Values from 3D MDS Coordinates for Experiment 3 (Non-composers, n = 9)

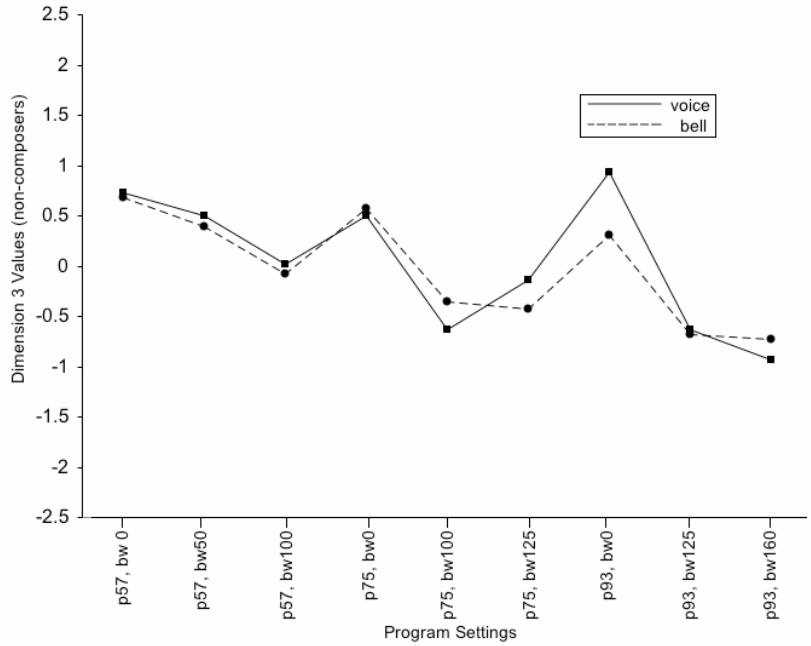
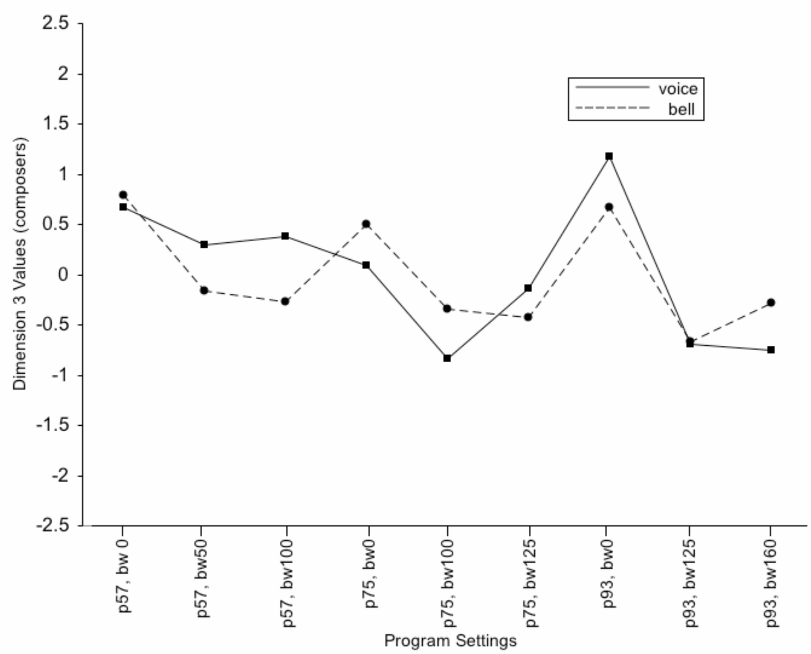


Figure 43. Dimension 3 Values from 3D MDS Coordinates for Experiment 3 (Composers, n = 13)



opposite direction of the grain length. This evidence supports a difference between the perceptions of randomization applied to grain length and grain period.

Graphs of the first dimension from each subgroup's 3D solution (see Figures 32 through 35) show the now familiar separation between the two sound sources. The second (see Figures 36 through 39) and third dimensions (see Figures 40 through 43) show modulation independent of sound sources, as was observed in results from the first two experiments. For both of these dimensions, only the non-listener subgroup (*n* = 5) looks remarkably different than the others (see Figures 36 & 40). However, the difference is largely attributable to values in these dimensions being inverted at zero along the y-axis. Beyond this the trends appear very similar across all subgroups. The three-segment shape that was observed in the previous results is not as pronounced as before, but does appear to be present within the third dimension. This visual analysis was less conclusive than in the previous experiments, but further analysis in the next chapter will aid in drawing better conclusions about the results and isolating the differences between the significant subgroups from this third experiment.

E. Conclusion

Overall the experiments successfully allowed the author to test his hypotheses. The low stress and high variance measures reported by the MDS algorithm showed that the patterns emerging from participants' responses were meaningful. Organizational features within the MDS solutions show that participants did perceive changes in the granular processing settings.

Participants clearly differentiated between the two sound sources used to produce stimuli. They were also able to perceive common settings used by the granular processing. The sound source became the feature of primary focus, likely due to the experiment design. The practice section for each experiment contained stimuli derived from only one sound source. The addition of two new sound sources in the recorded section may have lead participants to focus on this dichotomy. However, they also made separate distinctions on the basis of the program settings alone.

Visual analysis of the MDS solutions did not provide enough clarity to conclusively correlate parameter changes with the second and third dimensions. Participants in the first experiment did perceive differences in grain length and grain period, but in such a manner that these perceptions were commingled within the results. Randomization was perceived differently when applied to grain length and grain period, however the nature of this difference was unclear. It is possible that participants' responses reflected processing descriptors other than those used in the program that actually produced the stimuli, a hypothesis that will be explored in the next chapter.

Prior experience with electroacoustic music was not a significant factor among the participants of the first two experiments. In Experiment 3, the subgroups created by the operational definitions were significantly different. It was unclear how these differences were manifested in the subsequent MDS solutions. Participants' answers to the preliminary questions encompassed a large range, with large differences between the mean and median responses. Because of the variance in these statistical measures, it would not be prudent to make general conclusions about the effect prior experience with electroacoustic music has on the perception of granular processing. Any future experiments wishing to address this issue should employ a method different than the one from this study. Despite any issues surrounding the operational definitions, the subgroups from Experiment 3 will remain separate for any further analyses.

CHAPTER FIVE: FURTHER ANALYSIS OF RESULTS

The MDS solutions from Experiments 1, 2, and 3 successfully accounted for most of the variance in participants' responses. Although basic trends were identified by examining plotted graphs of the solution coordinates, the analysis could not identify the exact relationship between parameter settings and the subject's responses. In order to identify specific descriptors for each dimension, another form of analysis was needed. Some timbre studies that have employed MDS have also performed secondary acoustic analyses of the stimuli. Iverson and Krumhansl (1993) used this process to identify the relationships between centroid frequencies, amplitude envelopes and timbre discriminations during specific segments of musical tones. Kendall, Carterette and Hajda (1999) used the same process to identify the acoustical features responsible for discriminations between natural and synthetic instrument tones. The investigators first measured specific acoustical attributes for each stimulus, such as the centroid frequency. The measurements were then treated as independent variables and tested for correlation to the individual dimensions of each MDS solution. Identifying the significant correlations provided clearer evidence of the most salient feature in each dimension than visual analysis alone.

The acoustic measures tested by these studies were not as relevant to the current study as the parameters that have been applied to granular voices by various programmers (see chapter 3). As Grey (1977) pointed out, “The most direct and reliable information about the physical properties of stimuli is a knowledge of the parameters used for their synthesis" (p. 1271). Because these parameters were variations of one another, it was possible to develop a list of potential "descriptors" based on the program settings used to generate each stimulus. SPSS (Windows version 12.0) was used to compute the Pearson correlation between these descriptors and the five coordinate values produced for each stimulus by the MDS algorithm, two values from the 2D solution and three from the 3D solution. Descriptors with significant correlations to the individual dimensions could then be more confidently considered as possible control parameters for the new interface described in the final chapter. Nearly every descriptor correlated to the one of the five dimensions tested and multiple descriptors were significantly correlated to each dimension at the $p<.01$ level. Although these results provided clear evidence that variance in these dimensions was certainly coordinated with changes in the granular processing parameters, they also confirmed the aforementioned interrelated nature of potential descriptors. Significance alone was therefore an insufficient means of isolating the most appropriate descriptor for each dimension.

This chapter will highlight descriptors with the highest-recorded Pearson correlation for each dimension. These will be supplemented with tables displaying the recorded coefficients for each descriptor. In determining the highest correlation coefficients, comparisons will be made using the absolute value, because direction is an irrelevant attribute of MDS solutions. Results will be deemed "inconclusive" for any dimension in which more than two of the stimuli descriptors shared the highest absolute value for Pearson's correlation coefficient. This procedure will ensure the proper identification of only the most salient features.

A. Stimuli Descriptors

1. Actual Program Settings

Developing a list of potential descriptors (see Table 9) should begin with the control parameters used by the program that generated stimuli for the current study. The Max/MSP described in chapter 4 allowed for manipulations of grain length (L) and grain period (P), with both parameters expressed in milliseconds. Randomization was applied using a percentage deviation, representing the ratio between the linear deviation and the mean. This randomization was applied to length (LBW%) in Experiment 2 and to period (PBW%) in Experiment 3. These parameters were the most obvious descriptors for possible correlation with the MDS solution, because they were recorded directly from the program responsible for their creation. This fact may have provided them with an advantage over other descriptors correlated as part of this analysis, a potential that must be considered when interpreting the results.

Table 9. Summary of Descriptors Tested for Correlation with MDS Dimensions

		Measurement-based [b]	
Descriptors	Settings-based [a]	Experiment 2	Experiment 3
average grain length	L [c]	AL [c]	
average grain period	P [c]		AP [c]
grains per second	1000 / P		1000 / AP
grain delay	P – L	P – AL	AP – L
grain width	L / P	AL / P	L / AP
grain width • grains per second	(L / P) • (1000 / P)	(AL / P) • (1000 / P)	(L / AP) • (1000 / AP)
inverse of length	1000 / L	1000 / AL	
length bandwidth %	LBW% [c]	(L-max – L-min) / AL	
length bandwidth ms	LBW% • L	L-max – L-min	
minimum length	L – (0.5 • LBW% • L)	L-min [c]	
maximum length	L + (0.5 • LBW% • L)	L-max [c]	
period bandwidth %	PBW% [c]		(P-max – P-min) / AP
period bandwidth ms	PBW% • L		P-max – P-min
minimum period	P – (0.5 • PBW% • P)		P-min [c]
maximum period	P + (0.5 • PBW% • P)		P-max [c]
number of grains		NG [c]	NG [c]

[a] Descriptors in this column were based on the program settings used to produce stimuli. [b] Descriptors in these columns were based on values measured within the actual stimuli used within the experiments. [c] Denotes descriptors for which values were recorded directly. Formulas listed for the remaining descriptors express the manner in which these were calculated using the recorded values.

2. Literature-based Program Settings

The review of granular programs presented in chapter 3 provided a number of alternative descriptors to evaluate for significant correlation with the MDS solution. Values for all of these alternatives were computed using the actual program settings. Most of the programs profiled in Chapter 3 used density (also called grain frequency or grain rate) to control the rate of grains per second within a single voice. The grains per second for each stimulus was computed as 1000 ms divided by the grain period (1000 / P). Grain delay (Truax, 1988) was computed as the difference between the period and length settings (P – L). Grain width (Behles, Starke & Röbel, 1998) was computed as the length setting divided by the period setting (L / P).

Instead of a percentage of the mean value, some of the examples from Chapter 3 expressed random deviations in scalar values using the same unit of measurement as the mean (i.e., milliseconds for both grain length and grain period). The bandwidth percentage was used to compute a deviation descriptor based on the mean length in Experiment 2 (LBW% • L) and mean period in Experiment 3 (PBW% • P). Other examples constrained randomization using the maximum and minimum values instead of mean and bandwidth values. The respective bandwidth and mean settings from the second and third experiments were used to compute minimum (P – [0.5 • PBW% • P]) and maximum (L + [0.5 • LBW% • L]) descriptors.

3. Mathematically Related Descriptors

Several additional descriptors were developed using mathematical combinations and permutations of the descriptors previously mentioned. They were introduced as an attempt to capture the commingling of parameters observed in some of the graphed solutions. If dimensions of the MDS solutions did represent a blending of parameters, these descriptors might prove to be a plausible method of expressing their combination.

First, the operation used to compute the grains per second from the grain period was applied to the grain length (1000 / L). This will test for the possibility of an inverse to length as the basis for stimuli organization, just as grains per seconds acts as an inverse to grain period. Second, the product of the grain width ratio and grains per second ([L / P] • [1000 / P]) was computed to test for a hybrid of these parameters being represented. Recall that the first experiment's 3D MDS solution appeared to capture both length and period changes in its third dimension. If the perception of the two has truly commingled, then perhaps this combination will be found to provide an adequate description.

It was possible that descriptors would better correlate to the MDS solutions in a nonlinear fashion. The most plausible candidate for scaling certain parameters was determined to be a base-2 logarithmic relationship. The reasoning was based on the potential for amplitude envelopes to create effects in the frequency domain as the grain length is varied. Because doublings in frequency are equated with the perception of musical octaves, such doublings may have become a feature in the participants' perception of grain length. Similarly, the grain frequencies produced by the grain period may have activated the same perceptual feature, although such frequencies are below the range assigned to pitch. If this perceptual feature was apparent to participants, then the MDS solutions based on their evaluations of length and period changes may be organized according to these doublings. Base-2 logarithms were computed for grain length, grain period and grains per second, as well as the respective minimum and maximum descriptors of the constraints placed upon randomization.

4. Stimuli Measurements

The random deviations produced by the program for stimuli of finite length could have resulted in differences between the program settings and the actual grains produced for stimuli in Experiments 2 and 3. The mean and bandwidth program settings describe the sound output over long periods of time. Although they would have remained within the bounds created by these program settings, measurements of the actual mean and bandwidth may have differed from the settings during the relatively short amount of time that these stimuli lasted (1000 ms). Descriptors based on actual measurements could produce higher correlation values than descriptors derived from the program settings. Comparing such "measurements-based" descriptors to their "settings-based" counterparts could test for the importance of deviations from the program settings.

In order to obtain these measurements, each stimulus was examined with PEAK (version 3.2), an audio waveform editor. Markers were placed at the beginning and ending of each grain within the stimuli. Using PEAK's ability to display details at the level of individual digital audio samples, beginning markers were placed at the first sample with a non-zero amplitude value and ending markers at the last sample with a non-zero amplitude value. For stimuli from Experiment 2, the number of samples between each pair of beginning and ending markers was recorded, thereby measuring the length of each grain. For stimuli from Experiment 3, the number of samples between consecutive beginning markers was recorded, thereby measuring the period between each pair of grain onsets.

The mean of the recorded measurements for each stimulus was computed to produce the average length (AL) in Experiment 2 and average period (AP) in Experiment 3. Corresponding minimum (L-min or P-min) and maximum (L-max or P-max) values were also noted for each experiment. These three values were converted to milliseconds by dividing each by the sampling rate (44.1 samples per ms) and were then used to compute values for the other appropriate measurements-based descriptors. For Experiment 2, the length bandwidth in milliseconds (L-max – L-min), length bandwidth as a percentage of the mean ([L-max – L-min] / AL), grain width (AL / P) and average delay (P – AL) were computed. For Experiment 3, the period bandwidth in milliseconds (P-max – P-min), period bandwidth as a percentage of the mean ([P-max – P-min] / AP), grain width (L / AP), average delay (AP – L) and grains per second (1000 / AP) were computed. The mathematically-related descriptors detailed in the previous section were also computed again for each experiment using the appropriate measurements-based descriptors.

In addition to the measurements already described for Experiments 2 and 3, one final measurement was taken for each stimulus in all three experiments. Each stimulus was opened once again in the audio waveform editor and the number of grains (NG) was counted. This somewhat basic descriptor will be tested for correlation to the MDS solutions along with the other measurements-based descriptors.

Tables 10 and 11 list the average, minimum, maximum and number of grains values recorded for Experiments 2 and 3, respectively. While compiling these values for use with the correlation formula, the author noticed that the averages, minimums and maximums for certain stimuli were farther from the program settings than others, even occasionally falling outside the appropriate bounds. It was determined that the fade-out and fade-in for each stimulus had altered the length and period of the certain first and last grains, thereby altering the minimum values that would have been otherwise recorded. In order to

Table 10. Stimuli Measurements for Experiment 2

	Grain length (ms)			Number
Stimuli Reference	Average	Minimum	Maximum	of grains
b_l22_lbw0	22.02	22.00	22.04	12
b_l22_lbw125	21.83	11.38	34.58	12
b_l22_lbw160	18.06	8.53	34.67	12
b_l29_lbw0	29.00	29.00	29.02	12
b_l29_lbw100	25.77	17.32	34.26	12
b_l29_lbw125	32.73	21.59	45.87	12
b_l36_lbw0	36.01	36.01	36.01	12
b_l36_lbw50	36.43	29.91	42.02	12
b_l36_lbw100	36.31	22.47	46.67	12
v_l22_lbw0	22.02	22.02	22.04	12
v_l22_lbw125	22.67	9.34	32.49	11
v_l22_lbw160	21.73	6.46	30.75	11
v_l29_lbw0	29.00	29.00	29.00	11
v_l29_lbw100	29.98	17.80	40.29	11
v_l29_lbw125	27.66	18.05	39.18	12
v_l36_lbw0	36.01	36.01	36.03	11
v_l36_lbw50	35.89	29.00	43.51	12
v_l36_lbw100	34.02	18.89	43.70	12

Note. Stimuli References are comprised of the first letter of the sound source (b = bell, v = voice), program setting for grain length (ms) and program setting for length bandwidth (%).

Table 11. Stimuli Measurements for Experiment 3

	Grain period (ms)			Number
Stimuli Reference	Average	Minimum	Maximum	of grains
b_p57_pbw0	57.00	56.98	57.01	15
b_p57_pbw50	55.56	43.97	69.05	15
b_p57_pbw100	58.38	29.07	85.19	15
b_p75_pbw0	75.00	74.99	75.01	11
b_p75_pbw100	77.76	51.02	105.10	9
b_p75_pbw125	72.52	29.68	104.81	11
b_p93_pbw0	93.00	92.99	93.02	8
b_p93_pbw125	81.27	53.85	133.51	9
b_p93_pbw160	78.37	48.55	104.49	10
v_p57_pbw0	57.00	56.98	57.01	15
v_p57_pbw50	61.14	52.20	66.71	14
v_p57_pbw100	55.54	31.25	75.17	16
v_p75_pbw0	75.00	74.99	75.01	10
v_p75_pbw100	67.95	43.65	109.14	11
v_p75_pbw125	63.88	28.03	86.37	13
v_p93_pbw0	93.00	92.99	93.02	7
v_p93_pbw125	83.64	34.42	143.45	9
v_p93_pbw160	72.36	24.15	115.94	11

Note. Stimuli References are comprised of the first letter of the sound source (b = bell, v = voice), program setting for grain period (ms) and program setting for period bandwidth (%).

compensate for this, any measurements recorded for first and last grains from all stimuli in these two experiments were excluded from the records used to develop these tables. This added step caused the revised values to fall within expected deviations from the program settings.

B. Correlation Results

1. Experiment 1

Table 12 lists the Pearson correlations between each of the settings-based descriptors and the MDS dimensions for Experiment 1. Only the first dimension of both the 2D and 3D solutions failed to significantly correlate to any of the descriptors tested, something that was expected given the clear visual groupings based on sound source. All of the descriptors had a significant relationship with at least one of the remaining dimensions at the $p<.01$ level.

Descriptors associated with grain length had the highest correlations to the second dimensions of both solutions. In the 2D solution, both (1000 / L) and log2(L) correlated $r(16)=.91$, $p<.001$ and $r(16)=-.91$, $p<.001$, respectively, with the second dimension. This was slightly higher than L's correlation $r(16)=.89$, $p<.001$, indicating that differences in grain length were not rated on a linear scale by participants. The same was true of the 3D solution, where log2(L) had the highest correlation, $r(16)=.92$, $p<.001$, slightly more than both (1000 / L) and L, $r(16)=-.91$, $p<.001$ and $r(16)=.90$, $p<.001$, respectively. These results target

Table 12. Correlations Between Settings-based Descriptors and MDS Dimensions for Experiment 1

	2-D solution		3-D solution		
Stimuli descriptors	1	2	1	2	3
grain length (L) [a]	.16	.89**	.16	.90**	.14
log2(L)	.16	.91**	.16	.92**	.22
grain period (P) [a]	.17	.87**	.18	.86**	-.44
log2(P)	.17	.88**	.18	.87**	-.42
grains per second (1000 / P)	-.17	-.88**	-.18	-.87**	.40
log2(1000 / P)	-.17	-.88**	-.18	-.87**	.42
grain delay (P – L)	.11	.52*	.13	.51*	-.71**
grain width (L / P)	.06	.42	.05	.44	.71**
(L / P) • (1000 / P)	-.07	-.26	-.09	-.24	.83**
1000 / L	-.16	-.91**	-.16	-.91**	-.30
number of grains (NG) [b]	-.15	-.87**	-.17	-.86**	.38

Note. N = 18; This represents the number of stimuli used and not participants in the experiment. Correlation values are product-moment coefficients (Pearson r). The values used for each row were based on parameter settings from the program used to generate stimuli.

*Significant at the $p < .05$ level. ** Significant at the $p < .01$ level.

[a] L and P were the program parameters manipulated during stimuli generation. [b] NG represents a measurement taken by counting the number of grains while viewing each stimuli in an audio waveform editor.

length as the most salient feature and provide evidence that length was perceived on a logarithmic scale.

The third dimension had the highest correlation, $r(16)=.83$, $p<.001$, with the product of two descriptors, ([L / P] • [1000 / P]). This supports the earlier observation that differences related to length and period appear to have commingled within this dimension. The results of the second and third experiments will help isolate the effects of these parameters, because stimuli in those experiments limited program setting changes to either length or period.

2. Experiment 2

Tables 13 and 14 list the Pearson correlations for the second experiment's MDS dimension to the settings-based and measurement-based descriptors, respectively. The MDS dimensions from Experiment 2 showed significant correlations with every descriptor tested at the $p<.01$ level. The correlation coefficients improved when measurements were used instead of the program settings to develop descriptors, supporting the hypothesis that deviations from the program settings are important. The first dimension again failed to correlate with any descriptors in the 2D and 3D solutions, but did correlate $r(16)=.62$, $p<.01$, to the number of grains within each stimuli (NG). It is not known what conclusions, if any, can be drawn from this single correlation.

The second dimensions of the 2D and 3D solutions had the highest correlation with the settings-based log2(L – [0.5 • LBW% • L]), $r(16)=–.91$, $p<.001$

Table 13. Correlations Between Settings-based Descriptors and MDS Dimensions for Experiment 2

	2-D solution		3-D solution		
Stimuli descriptors	1	2	1	2	3
grain length (L) [a]	.06	-.81**	.06	-.80**	.49*
log2(L)	.06	-.81**	.06	-.81**	.49*
grain period (P) [b]					
grains per second (1000 / P) [b]					
grain delay (P – L)	-.06	.81**	-.06	.80**	-.49*
grain width (L / P)	.06	-.81**	.06	-.80**	.49*
(L / P) • (1000 / P)	.06	-.81**	.06	-.80**	.49*
1000 / L	-.06	.82**	-.06	.82**	-.49*
L bandwidth % (LBW%) [a]	-.09	.71**	-.09	.73**	.49*
L bandwidth ms (LBW% • L)	-.08	.51*	-.08	.53*	.59*
minimum length (L – [0.5 • LBW% • L])	.09	-.86**	.09	-.87**	-.17
log2(L – [0.5 • LBW% • L])	.10	-.91**	.09	-.93**	-.19
maximum length (L + [0.5 • LBW% • L])	-.03	-.08	-.03	-.06	.80**
log2(L + [0.5 • LBW% • L])	-.04	-.05	-.03	-.03	.83**

Note. N = 18. This represents the number of stimuli used and not participants in the experiment. Correlation values are product-moment coefficients (Pearson r). The values used for each row were based on parameter settings from the program used to generate stimuli.

*Significant at the p < .05 level. ** Significant at the p < .01 level.

[a] L and LBW% were the program parameters manipulated during stimuli generation. [b] P (and therefore the grains per second) did not vary between stimuli and could not be correlated to the MDS solutions.

Table 14. Correlations Between Measurement-based Descriptors and MDS Dimensions for Experiment 2

	2-D solution		3-D solution		
Stimuli measurements	1	2	1	2	3
number of grains (NG)	.62**	-.04	.63**	-.04	-.14
average length (AL)	.06	-.84**	.05	-.84**	.46
log2(AL)	.03	-.86**	.03	-.86**	.44
average delay (P – AL) [a]	-.06	.84**	-.05	.84**	-.46
grain width (AL / P) [a]	.06	-.84**	.05	-.84**	.46
(AL / P) • (1000 / P) [a]	.06	-.84**	.05	-.84**	.46
1000 / AL	-.00	.87**	-.00	.87**	-.41
L bandwidth % ([L-max – L-min] / AL)	-.11	.57*	-.10	.59*	.60**
L bandwidth ms (L-max – L-min)	-.08	.79**	-.08	.81**	.46
minimum length (L-min)	.17	-.90**	.17	-.91**	-.09
log2(L-min)	.19	-.92**	.19	-.93**	-.11
maximum length (L-max)	.05	-.25	.06	-.23	.80**
log2(L-max)	.04	-.21	.04	-.20	.82**

Note. $N = 18$. This represents the number of stimuli used and not participants in the experiment. Correlation values are product-moment coefficients (Pearson *r*). The values used for each row were measurements recorded by observing the actual stimuli within an audio waveform editor.

*Significant at the $p < .05$ level. ** Significant at the $p < .01$ level.

[a] P values were those from the actual program settings and not measurements.

and $r(16)=-.93$, $p<.001$, respectively. The third dimension of the 3D solution had the highest correlation with the settings-based log2(L + [0.5 • LBW% • L]), $r(16)=.80$, $p<.001$. Coefficients improved slightly when the corresponding measurements-based descriptors were correlated. The second dimensions correlated $r(16)=-.92$, $p<.001$ and $r(16)=-.93$, $p<.001$, respectively, with log2(L-min). The third dimension correlated $r(16)=.82$, $p<.001$, to log2(L-max).

These results support the earlier hypothesis that the base-2 logarithm is a perceptually meaningful formula for expressing the relationship between various grain lengths. Correlation of the MDS solution dimensions to this non-linear relationship strengthen the hypothesis that doublings in duration are a salient feature of granular processing. These results also identify minimum and maximum values as the most salient features when organizing randomized grain lengths. This runs counter to the program settings used to generate these stimuli, removing any question about a possible advantage that the mean and bandwidth descriptors had over others.

3. Experiment 3

Descriptors for Experiment 3 were correlated to the MDS solution coordinates for each of the operationally defined subgroups separately because of the significant differences found via MANOVA. Within every solution produced for this experiment the first dimensions failed to significantly correlate with any of the descriptors tested, reaffirming the earlier conclusion that all MDS solutions produced for this study placed the sound source dichotomy within Dimension 1. As with the first two experiments, significant correlations were confined to the second and third dimensions reaffirming their association with changes in the granular processing.

Tables 15, 16, 17, and 18 list the correlation coefficients between the dimensions for each subgroup's 2D and 3D solutions and the settings-based descriptors. Most of the stimuli descriptors correlated at the $p<.01$ level to at least one dimension from each of the subgroup solutions. Equivalent correlations to the second dimensions of all solutions meant that this portion of the analysis was inconclusive. The inability to identify a "best" descriptor was somewhat surprising given the success of prior analyses, but could be attributable to the aforementioned close relationships between descriptors. Dimension 3 values from the "non-listener" subgroup had the highest correlation with (P – [0.5 • PBW% • P]), $r(16)=-.80$, $p<.001$, while values from "listener" subgroup correlated $r(16)=-.81$, $p<.001$, instead to (PBW% • P). The third dimension for the "non-composer" had its highest correlation with (PBW% • P), $r(16)=-.95$, $p<.001$, while this dimension from the "composer" subgroup solution was equally correlated $r(16)=-.82$, $p<.001$, to PBW% and (PBW% • P).

Tables 19, 20, 21, and 22 list the Pearson correlations between the measurements-based descriptors and the MDS dimensions from each subgroup. Only one dimension from a single subgroup produced inconclusive results, which

Table 15. Correlations Between Settings-based Descriptors and MDS Dimensions for Experiment 3 (Non-listeners, n = 5)

	2-D solution		3-D solution		
Stimuli descriptors	1	2	1	2	3
grain length (L) [a]					
grain period (P) [b]	.06	.75**	.07	.86**	-.02
log2(P)	.05	.75**	.06	.87**	-.02
grains per second (1000 / P) [b]	-.05	-.75**	-.06	-.87**	.03
log2(1000 / P)	-.05	-.75**	-.06	-.87**	.02
grain delay (P – L)	.06	.75**	.07	.86**	-.02
grain width (L / P)	-.05	-.75**	-.06	-.87**	.03
(L / P) • (1000 / P)	-.05	-.75**	-.06	-.86**	.03
P bandwidth % (PBW%) [a]	-.05	.10	-.04	.44	.74**
P bandwidth ms (PBW% • P)	-.04	.18	-.03	.53*	.70**
minimum period (P – [0.5 • PBW% • P])	.08	.27	.08	-.04	-.80**
log2(P – [0.5 • PBW% • P])	.07	.14	.06	-.18	-.76**
maximum period (P + [0.5 • PBW% • P])	-.01	.45	.00	.75**	.51*
log2(P + [0.5 • PBW% • P])	-.01	.49*	.01	.79**	.48*

Note. *N* = 18. This represents the number of stimuli used and not participants in the experiment. Correlation values are product-moment coefficients (Pearson *r*). The values used for each row were based on parameter settings from the program used to generate stimuli.

*Significant at the $p < .05$ level. ** Significant at the $p < .01$ level.

[a] P and PBW% were the program parameters manipulated during stimuli generation. [b] L did not vary between stimuli and could not be correlated to the MDS solutions.

Table 16. Correlations Between Settings-based Descriptors and MDS Dimensions for Experiment 3 (Listeners, n = 17)

	2-D solution		3-D solution		
Stimuli descriptors	1	2	1	2	3
grain length (L) [a]					
grain period (P) [b]	.04	.80**	.05	-.79**	-.41
log2(P)	.04	.80**	.05	-.79**	-.43
grains per second (1000 / P) [b]	-.04	-.79**	-.04	.79**	.45
log2(1000 / P)	-.04	-.80**	-.05	.79**	.43
grain delay (P – L)	.04	.80**	.05	-.79**	-.41
grain width (L / P)	-.04	-.79**	-.04	.79**	.45
(L / P) • (1000 / P)	-.04	-.78**	-.04	.78**	.46
P bandwidth % (PBW%) [a]	-.01	-.06	.00	.08	-.80**
P bandwidth ms (PBW% • P)	.00	.03	.01	-.01	-.81**
minimum period (P – [0.5 • PBW% • P])	.03	.47*	.02	-.49*	.65**
log2(P – [0.5 • PBW% • P])	.02	.34	.01	-.36	.66**
maximum period (P + [0.5 • PBW% • P])	.02	.36	.03	-.34	-.77**
log2(P + [0.5 • PBW% • P])	.02	.42	.03	-.40	-.78**

Note. *N* = 18. This represents the number of stimuli used and not participants in the experiment. Correlation values are product-moment coefficients (Pearson *r*). The values used for each row were based on parameter settings from the program used to generate stimuli.

*Significant at the $p < .05$ level. ** Significant at the $p < .01$ level.

[a] P and PBW% were the program parameters manipulated during stimuli generation. [b] L did not vary between stimuli and could not be correlated to the MDS solutions.

Table 17. Correlations Between Settings-based Descriptors and MDS Dimensions for Experiment 3 (Non-composers, n = 9)

	2-D solution		3-D solution		
Stimuli descriptors	1	2	1	2	3
grain length (L) [a]					
grain period (P) [b]	.03	-.76**	.04	-.75**	-.47*
log2(P)	.03	-.76**	.04	-.75**	-.48*
grains per second (1000 / P) [b]	-.03	.76**	-.04	.75**	.48*
log2(1000 / P)	-.03	.76**	-.04	.75**	.48*
grain delay (P – L)	.03	-.76**	.04	-.75**	-.47*
grain width (L / P)	-.03	.76**	-.04	.75**	.48*
(L / P) • (1000 / P)	-.03	.75**	-.03	.74**	.48*
P bandwidth % (PBW%) [a]	-.05	.10	-.05	.13	-.93**
P bandwidth ms (PBW% • P)	-.04	-.00	-.04	.03	-.95**
minimum period (P – [0.5 • PBW% • P])	.06	-.49*	.07	-.51*	.77**
log2(P – [0.5 • PBW% • P])	.05	-.36	.05	-.39	.81**
maximum period (P + [0.5 • PBW% • P])	-.01	-.32	-.01	-.29	-.89**
log2(P + [0.5 • PBW% • P])	-.01	-.37	-.01	-.34	-.89**

Note. N = 18. This represents the number of stimuli used and not participants in the experiment. Correlation values are product-moment coefficients (Pearson r). The values used for each row were based on parameter settings from the program used to generate stimuli.

*Significant at the $p < .05$ level. ** Significant at the $p < .01$ level.

[a] P and PBW% were the program parameters manipulated during stimuli generation. [b] L did not vary between stimuli and could not be correlated to the MDS solutions.

Table 18. Correlations Between Settings-based Descriptors and MDS Dimensions for Experiment 3 (Composers, n = 13)

	2-D solution		3-D solution		
Stimuli descriptors	1	2	1	2	3
grain length (L) [a]					
grain period (P) [b]	.06	.83**	.06	-.85**	-.26
log2(P)	.06	.82**	.06	-.85**	-.28
grains per second (1000 / P) [b]	-.06	-.82**	-.06	.85**	.29
log2(1000 / P)	-.06	-.82**	-.06	.85**	.28
grain delay (P – L)	.06	.83**	.06	-.85**	-.26
grain width (L / P)	-.06	-.82**	-.06	.85**	.29
(L / P) • (1000 / P)	-.05	-.81**	-.06	.84**	.31
P bandwidth % (PBW%) [a]	.01	.00	.01	-.05	-.82**
P bandwidth ms (PBW% • P)	.01	.10	.02	-.14	-.82**
minimum period (P – [0.5 • PBW% • P])	.02	.42	.02	-.38	.75**
log2(P – [0.5 • PBW% • P])	.01	.28	.00	-.24	.73**
maximum period (P + [0.5 • PBW% • P])	.03	.41	.04	-.46	-.71**
log2(P + [0.5 • PBW% • P])	.04	.47*	.04	-.52*	-.72**

Note. *N* = 18. This represents the number of stimuli used and not participants in the experiment. Correlation values are product-moment coefficients (Pearson *r*). The values used for each row were based on parameter settings from the program used to generate stimuli.

*Significant at the $p < .05$ level. ** Significant at the $p < .01$ level.

[a] P and PBW% were the program parameters manipulated during stimuli generation. [b] L did not vary between stimuli and could not be correlated to the MDS solutions.

Table 19. Correlations Between Measurement-based Descriptors and MDS Dimensions for Experiment 3 (Non-listeners, n = 5)

	2-D solution		3-D solution		
Stimuli measurements	1	2	1	2	3
number of grains (NG)	-.11	-.81**	-.12	-.82**	.27
average period (AP)	.16	.80**	.17	.79**	-.31
log2(AP)	.15	.80**	.16	.80**	-.28
grains per second (1000 / AP)	-.14	-.80**	-.15	-.81**	.25
log2(1000 / AP)	-.15	-.80**	-.16	-.80**	.28
average delay (AP – L) [a]	.16	.80**	.17	.79**	-.31
grain width (L / AP)	-.14	-.80**	-.15	-.81**	.25
(L / AP) • (1000 / P) [a]	-.13	-.79**	-.14	-.82**	.23
P bandwidth % ([P-max – P-min] / AP)	-.14	.04	-.13	.40	.77**
P bandwidth ms (P-max – P-min)	-.13	.11	-.11	.47*	.75**
minimum period (P-min)	.21	.39	.20	.10	-.79**
log2(P-min)	.24	.30	.23	.02	-.74**
maximum period (P-max)	-.01	.50*	.01	.78**	.43
log2(P-max)	.02	.56*	.03	.82**	.39

Note. *N* = 18. This represents the number of stimuli used and not participants in the experiment. Correlation values are product-moment coefficients (Pearson *r*). The values used for each row were measurements recorded by observing the actual stimuli within an audio waveform editor.

*Significant at the $p < .05$ level. ** Significant at the $p < .01$ level.

[a] L values were those from the actual program settings and not measurements.

Table 20. Correlations Between Measurement-based Descriptors and MDS Dimensions for Experiment 3 (Listeners, n = 17)

	2-D solution		3-D solution		
Stimuli measurements	1	2	1	2	3
number of grains (NG)	-.11	-.88**	-.11	.88**	.29
average period (AP)	.14	.93**	.14	-.93**	-.16
log2(AP)	.14	.91**	.14	-.91**	-.21
grains per second (1000 / AP)	-.13	-.89**	-.13	.89**	.26
log2(1000 / AP)	-.14	-.91**	-.14	.91**	.21
average delay (AP – L) [a]	.14	.93**	.14	-.93**	-.16
grain width (L / AP)	-.13	-.89**	-.13	.89**	.26
(L / AP) • (1000 / P) [a]	-.12	-.87**	-.13	.87**	.31
P bandwidth % ([P-max – P-min] / AP)	-.09	-.06	-.08	.08	-.83**
P bandwidth ms (P-max – P-min)	-.07	.03	-.06	-.00	-.85**
minimum period (P-min)	.15	.54*	.15	-.56*	.57*
log2(P-min)	.19	.43	.18	-.45	.56*
maximum period (P-max)	.03	.50*	.04	-.49*	-.76**
log2(P-max)	.05	.55*	.06	-.53*	-.77**

Note. *N* = 18. This represents the number of stimuli used and not participants in the experiment. Correlation values are product-moment coefficients (Pearson *r*). The values used for each row were measurements recorded by observing the actual stimuli within an audio waveform editor.

*Significant at the $p < .05$ level. ** Significant at the $p < .01$ level.

[a] L values were those from the actual program settings and not measurements.

Table 21. Correlations Between Measurement-based Descriptors and MDS Dimensions for Experiment 3 (Non-composers, n = 9)

	2-D solution		3-D solution		
Stimuli measurements	1	2	1	2	3
number of grains (NG)	-.10	.87**	-.10	.86**	.25
average period (AP)	.14	-.90**	.14	-.90**	-.16
log2(AP)	.13	-.88**	.13	-.88**	-.21
grains per second (1000 / AP)	-.12	.86**	-.13	.86**	.25
log2(1000 / AP)	-.13	.88**	-.13	.88**	.21
average delay (AP – L) [a]	.14	-.90**	.14	-.90**	-.16
grain width (L / AP)	-.12	.86**	-.13	.86**	.25
(L / AP) • (1000 / P) [a]	-.11	.84**	-.12	.84**	.28
P bandwidth % ([P-max – P-min] / AP)	-.14	.10	-.13	.13	-.90**
P bandwidth ms (P-max – P-min)	-.12	.01	-.11	.04	-.93**
minimum period (P-min)	.19	-.56*	.19	-.59*	.65**
log2(P-min)	.23	-.47	.22	-.49*	.64**
maximum period (P-max)	-.01	-.47*	-.01	-.45	-.81**
log2(P-max)	.01	-.51*	.02	-.48*	-.81**

Note. *N* = 18. This represents the number of stimuli used and not participants in the experiment. Correlation values are product-moment coefficients (Pearson *r*). The values used for each row were measurements recorded by observing the actual stimuli within an audio waveform editor.

*Significant at the $p < .05$ level. ** Significant at the $p < .01$ level.

[a] L values were those from the actual program settings and not measurements.

Table 22. Correlations Between Measurement-based Descriptors and MDS Dimensions for Experiment 3 (Composers, n = 13)

	2-D solution		3-D solution		
Stimuli measurements	1	2	1	2	3
number of grains (NG)	-.12	-.88**	-.13	.89**	.12
average period (AP)	.16	.92**	.16	-.93**	.01
log2(AP)	.15	.91**	.16	-.92**	-.04
grains per second (1000 / AP)	-.15	-.89**	-.15	.91**	.09
log2(1000 / AP)	-.15	-.91**	-.16	.92**	.04
average delay (AP – L) [a]	.16	.92**	.16	-.93**	.01
grain width (L / AP)	-.15	-.89**	-.15	.91**	.09
(L / AP) • (1000 / P) [a]	-.14	-.87**	-.14	.89**	.13
P bandwidth % ([P-max – P-min] / AP)	-.08	.00	-.07	-.05	-.87**
P bandwidth ms (P-max – P-min)	-.05	.08	-.05	-.13	-.88**
minimum period (P-min)	.15	.48*	.15	-.45	.69**
log2(P-min)	.19	.36	.18	-.33	.65**
maximum period (P-max)	.05	.54*	.06	-.58*	-.70**
log2(P-max)	.07	.58*	.08	-.62**	-.70**

Note. N = 18. This represents the number of stimuli used and not participants in the experiment. Correlation values are product-moment coefficients (Pearson r). The values used for each row were measurements recorded by observing the actual stimuli within an audio waveform editor.

*Significant at the $p < .05$ level. ** Significant at the $p < .01$ level.

[a] L values were those from the actual program settings and not measurements.

made the measurements-based descriptors more successful than their settings-based counterparts at isolating the most appropriate labels. The second dimension of the 2D solution for the non-listener subgroup produced the highest correlation $r(16)=-.81$, $p<.001$, with the number of grains. However, the results for Dimension 2 within the 3D solution were inconclusive. The non-listener subgroup's third dimensions had the highest correlation $r(16)=-.79$, $p<.001$, with the descriptor P-min, just as it did when the settings-based descriptors were tested. The second dimension of both listener subgroup solutions correlated (2D: $r(16)=.93$, $p<.001$ & 3D: $r(16)=-.93$, $p<.001$) at its highest level with AP and (AP – L), while the third dimension correlated $r(16)=-.85$, $p<.001$, at its highest level with (P-max – P-min). The measurements-based descriptors AP and (AP – L) produced the highest correlations with the second dimension values for both the non-composer (2D: $r(16)=-.90$, $p<.001$ & 3D: $r(16)=-.90$, $p<.001$) and composer (2D: $r(16)=.92$, $p<.001$ & 3D: $r(16)=-.93$, $p<.001$) subgroups. The descriptor (P-max – P-min) had the highest correlation to the third dimension values generated by the non-composer and composer subgroups, $r(16)=-.93$, $p<.001$ and $r(16)=-.88$, $p<.001$, respectively.

The difference in which descriptors resulted in the highest correlations for the non-listener (n = 5) and listener (n = 17) solutions confirms the earlier significant differences found via the MANOVA. The number of grains and the minimum period were the most salient features for the non-listener subgroup, while salient features for the listener subgroup included average period, average delay and the period bandwidth. The non-composer (*n* = 9) and composer (*n* = 13) subgroups exhibited no difference in the specific descriptors that were most salient. These descriptors were the same as those identified for the listener subgroup and differed only in their level of correlation.

It is important to remember that the primary goal of this study was not to identify differences between subgroups, but to inform the design of a new interface for granular processing. Therefore, the author must determine which subgroup's results from Experiment 3 are most pertinent. The non-listener subgroup (*n* = 5) represents a minority among the participants (*N* = 22). According to the reasoning that prompted the consideration of subgroups, non-listeners were less likely than listeners to have previously heard granular sounds. It is logical then to presume non-listeners would also be less likely to use a program for producing these sounds. The correlation results for the listener subgroup have additional pertinence because of the similarity to those for the composer and non-composer subgroups. Therefore, preference will be given to correlation results from these subgroups over those from the non-listeners when designating descriptors for control parameters within the new program interface.

For the preferred subgroups, grain period and grain delay were the most salient features, with both equally correlated to the second dimension. Isolating a preference for one over the other in this experiment is not possible because the two descriptors were exactly correlated $r(16)=1.00$, $p<.001$ to one another. A similar situation arose in Experiment 2, where grain length was exactly correlated $r(16)=1.00$, $p<.001$ to grain delay, and were therefore both equally correlated to the MDS dimensions (see Tables 13 & 14). In order to determine grain delay's importance as a descriptor, one must consider the results of Experiment 1. In that experiment, grain delay correlated $r(16)=.84$, $p<.001$ to grain period and did not correlate $r(16)=.22$, $p>.05$ to grain length. While none of these were identified as the most salient feature, grain delay did produce the weakest correlations to the MDS dimensions among the three (see Table 12). The individual experiment results provide insufficient evidence to determine the importance of grain delay as descriptor, but collectively they suggest that it is not as salient as either grain period or grain length.

The emergence of period bandwidth as the most salient feature of the third dimension is an important result from the third experiment. It is evidence of a key difference in the perception of randomization when applied to specific processing parameters. Experiment 3 showed that average and bandwidth were salient features of randomized grain periods. This is different from Experiment 2, where the minimum and maximum were salient features of randomized grain lengths. In addition, the bandwidth in milliseconds was more salient than the bandwidth as a percentage of the mean, dispelling any remaining concerns about an advantage that actual program settings had over the other descriptors.

C. Summary

The correlation of MDS dimensions to specific descriptors affirmed that participants' responses to stimuli pairings were indeed directly related to changes in the processing settings. The correlation coefficients provided sufficient evidence to form conclusions about the best label for each dimension. Changes in the grain's length were clearly a salient feature of these MDS solutions. The base-2 logarithmic descriptor was a better fit than the linear description in milliseconds, indicating that doublings in length were salient for participants. The minimum and maximum values provided the best description of grain length randomization. The grain period measured in milliseconds was found to be a better descriptor than either grains per second or grain delay for the organization of consecutive grains. When randomized, the average period was a salient feature. Deviations in the grain period were best described by the total bandwidth expressed in milliseconds. The descriptors identified in this chapter provide more definitive labels for the MDS solutions and will form the basis for the author's new granular processing interface.

CHAPTER SIX: APPLICATION OF FINDINGS

The theory proposed by Gabor (1947) and its musical interpretation by Xenakis (1963/1992) laid a firm foundation for granular techniques. After Roads (1978) and Truax (1986b) completed their early experiments to implement computer programs based on this foundation, an assortment of software for producing granular sounds was developed (see chapter 3). During this period, granular techniques have moved to "the forefront of compositional interest" (Roads, 2001, p. 21) among practitioners of experimental computer music. As experimental composers sought to transform their programming work into normative tools, they most often took the "confrontational approach" (Zicarelli, 2002) to interface design in an effort to avoid limiting users' actions. Such confrontational interfaces provided control over every aspect of the underlying granular algorithm and as a result exacerbated the steep learning curve often associated with granular techniques.

A series of three experiments were conducted to provide empirical evidence of the most salient features in granular sounds and inform the development of prudent limitations for a simplified granular processing interface. The experiments were designed to utilize multidimensional scaling (MDS), an analysis technique that had previously been proven useful for empirical research that informs computer music software design (Wessel, 1979). To generate stimuli for each experiment, a series of nine specific settings were used to process two different sound sources, a ringing bell and female singing voice. Because the current investigation focused on the perception of changes in grain duration and voice organization, as well as randomization of these two parameters, variations between processing settings were restricted to the relevant parameters of grain length and grains per second.

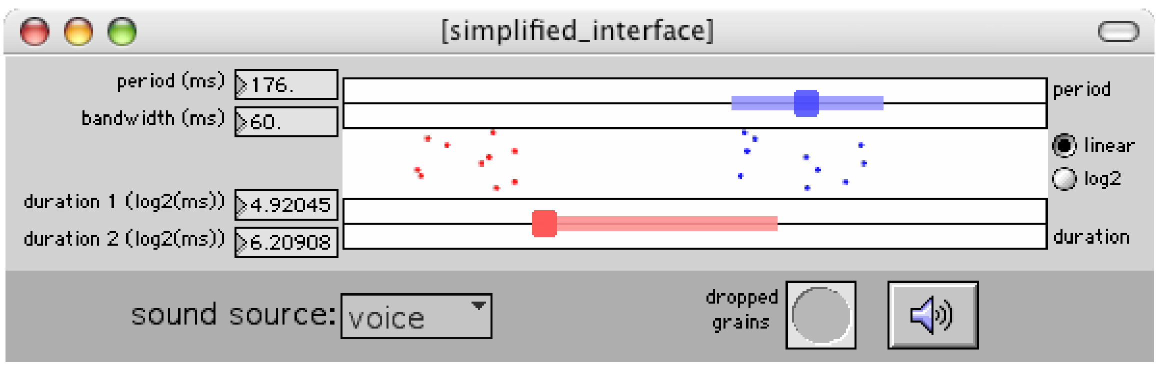
Participants heard all 171 unique stimuli pairings in a random order and provided a similarity rating for each. These ratings were averaged together and used to develop 2D and 3D MDS solutions. The first dimension in all resulting solutions exhibited an organization attributed to differences in sound source. The MDS coordinates from the remaining dimensions were treated as dependent variables and tested for correlation to independent variables derived from the granular processing settings. This analysis was used to determine the best descriptor for differences perceived by participants in responses to manipulations of the granular parameters. Changes in grain period expressed in milliseconds and scaled linearly were found to be best descriptor for differences in response to manipulations of the grains per second. Changes in grain duration expressed in milliseconds and scaled using a base-2 logarithmic function were found to be the best descriptor for differences in response to manipulations of the grain length. Randomizations produced different salient features for each of these processing descriptors, the mean and bandwidth for grain period and the minimum and maximum for grain duration.

To facilitate a secondary inquiry into the role of experience with electroacoustic music on the perception of granular sounds, participants were asked questions about their prior listening and composing experience. Responses were used to divide participants into subgroups based on operational definitions. Significant differences were found between subgroups in only the third experiment, but participants' responses to the experience questions exhibited a high degree of statistical variation overall. This was likely evidence that their interpretations of what constituted "electroacoustic music" were highly individual, despite being given a single definition prior to questioning. Therefore, no conclusions related to this secondary inquiry were offered.

A. Interface Design

The author returned to Max/MSP (version 4.5.2 for Mac OS X) to develop a new interface for granular processing (see Figure 44) based on the results presented in chapters 4 and 5. This interface was a demonstration of the effectiveness of empirical research in solving program design issues. Because the findings could not answer every design question, judicious interpretation was necessary to develop a workable GUI. The description of this process has been deliberately separated from the results in the previous chapter so that the distinction between them is clear. This section will provide an explanation of the new interface's innovative elements and account for the reasoning behind their

Figure 44. Screenshot from the author's new granular processing interface



design and inclusion. The unique interface elements described in this chapter were programmed using an implementation of the JavaScript language for Max/MSP (Cycling74, 2004; see Appendix B).

1. Control

Among the fifteen programs surveyed in chapter 3, eight used GUIs to control the underlying processing algorithm. Four of these eight programs used sliders to control some element of the processing (Behles, Starke, & Röbel, 1998; Nelson, 2000; Tanghe, 2003; Roads & Alexander, 1997). Sliders are a basic GUI element used to control parameter values within a predefined range. The length of a slider graphically represents the permissible range with the two endpoints corresponding to the minimum and maximum values allowed. The position of the handle provides a visual reference of how the current value relates to the permissible range. A typical slider has a single handle that the user may click and drag to any position along its single horizontal or vertical dimension, resulting in corresponding changes to the associated parameter. Within the author's new interface, horizontal sliders were used to control grain duration and period. This choice was made because of the features highlighted here and the familiarity users are likely to have with their operation. However, the sliders' basic construction was modified to reflect the specific perceptual features identified by the empirical study.

For grain duration, the maximum and minimum values that bound randomization must be controllable. Existing programs have handled this situation by having two separate controls, one for each value. Tanghe (2003) used proximity to visually group the minimum and maximum sliders together so that users would identify the common processing attribute. To eliminate the need for two controls, the author developed a single, horizontal slider for controlling both the minimum and maximum bounds. This new GUI element retains a single handle, but uses the mouse's ability to move in two dimensions to control both minimum and maximum values. Horizontal and vertical movements can change these values independently, while diagonal movements can alter both with a single click and drag motion. The handle represents the minimum value to be used, with horizontal mouse movements controlling its placement. A thin band extends from the right side of the handle, the length of which corresponds to the range of possible grain duration values. The right end of this band represents the maximum value, with vertical mouse movements controlling its placement. Changes to these minimum and maximum duration controls are scaled using a base-2 logarithmic function with randomizations between them distributed uniformly along the same scale.

For grain period, the mean and bandwidth values used to constrain randomization must be controllable. Just as they did for maximum and minimum bounds, existing programs have handled this situation by having two separate controls. In the case of *Stampede II* (Behles, Starke, & Röbel, 1998), columns were used to group sliders controlling the average value and amount of random deviation. Building on the design philosophy that guided the duration control, the author developed a second slider capable of controlling both the mean period and randomization bandwidth through a single handle. Again, horizontal and vertical movements can change these values independently, while diagonal movements can alter both. The handle on this new GUI element represents the mean value. When the user clicks and drags this handle, horizontal movements result in changes to the mean. A thin band extends from the handle in both directions, providing a visual representation of the randomization bandwidth. When the user clicks and drags the handle in a vertical direction, upward movements increase the bandwidth and downward movements decrease it. The band behind the handle expands and contracts to match the adjustment in period bandwidth. The user's movements create corresponding linear changes to the number of milliseconds between each grain onset, with randomized values uniformly distributed along the same scale.

2. Feedback

Although these sliders provide an effective means of controlling the underlying algorithm, they do not reflect the random variations between individual grains heard in the program's output. The empirical findings of this study provided evidence that participants' perception was closer to the actual audio output than the program settings. To aid in the users' ability to compare the two, the interface should provide users with a visual representation of these differences. In addition, because these new sliders use different scales to control their respective parameters, they give a false impression of the relationship between the duration and period settings when placed next to each other. The interface must provide another means of conveying the true relationship between these settings. To address both of these issues, the author developed a horizontal feedback display to be placed between the two control sliders.

Within the display area, there are twenty points to display the actual values used by the processing. Half of these points represent the last ten duration values, while the other half represents the last ten period values. New values are added to the bottom of the display as the existing points move up one position, giving the appearance that the points scroll upward. Color was used to distinguish between the two sets of points. The points representing duration are red, while the points representing period are blue. Different hues of these colors were used for the corresponding slider handles and bands, a simple and effective design strategy to visually group elements of the feedback display to the corresponding control sliders.

In order to accurately convey the relationship between duration and period values, the feedback display must use a single scale to plot both. However, because the control sliders each use a different scaling function, linear for period and base-2 logarithmic for duration, limiting the display to just one of these would give the appearance of a discrepancy between the feedback and one of the control sliders. To avoid this limitation, a set of radio buttons placed next to the display allows users to select their preferred scale and dynamically change the scaling function used to plot feedback. This dynamic scaling option couples with the static scaling of the control sliders to provide an overall interface that meets the design challenges posed by the empirical evidence.

3. Additional GUI Elements

A few additional elements were needed in order to make the program a usable demonstration of this study's empirical findings. The most obvious of these is a means to turn processing on and off. The button used is a common element from the Max/MSP environment containing an icon of a speaker. After the user clicks on the button, the graphic changes to appear "depressed", while processing begins and sound is produced. When the user clicks it again, the button returns to a "raised" look as processing and sound output end. The author chose to use this common button element because of its familiarity to Max/MSP users.

Controls for parameters not examined by the current study (e.g., pitch controls, envelope shape) were excluded from the interface in order to focus this demonstration specifically on the empirical findings. Values for these excluded parameters are fixed at constant values similar to those used to produce stimuli.

The pitch multiplier is set at 1.0 with no randomization. A Gaussian-shaped amplitude envelope is applied to each grain produced by the underlying algorithm. No randomization is applied to the sample offset. Users may select between the three sound sources used in this study's experiments: noise, bell and female voice.

To inform users when their settings tax the algorithm beyond its limits, the program monitors the processing load. The underlying algorithm is only capable of maintaining eight grains simultaneously, a capability that may be exceeded at certain interface settings. The program signals a warning indicator labeled "dropped grains" to flash whenever the eight grains capacity is exceeded. Although the algorithm will continue to output sound while the interface settings remain beyond its limitations, this indicator ensures that users do not assume the sound output is always an accurate reflection of the current program settings and informs them when incongruities occur. This additional feedback display based on a similar interface element found in the *KTGranulator* (Tanghe, 2003).

B. Reflection and Future Considerations

After investing the time and energy required to conduct these experiments, it is necessary to reflect on the value of their contribution to the software development process. Based on this experience, it is the author's opinion that empirical research methods are a valuable tool that more computer music programmers should consider employing. The findings of these experiments have led to an interface design that would likely have not been imaginable otherwise. The author began this study with his own intuitions about the perceptually salient features of granular processing, like several of the programmers profiled in chapter 3. However, the findings countered nearly all of the author's earlier intuition-based parameters (Wolek, 2001), pointed to new possibilities and provided sound evidence to support multiple interface changes.

Possessing such evidence is important, but it quickly becomes moot if the subsequent revisions do not improve the interface. After using the new implementation, the author can objectively report that the interface enhancements do offer clear improvements. The base-2 logarithmic scale offers smoother transitions and a more continuous feel than the linear scale for changes in grain duration. In addition, using randomized duration values that are distributed uniformly along this same logarithmic scale gives the sound output a texture that remains dynamic, but is less erratic than a linear distribution. The change in scaling function for duration is arguably the biggest innovation to result from this study. In addition, the use of maximum and minimum values as constraints upon duration randomizations seems particularly suited for this scaling function. The base-2 logarithmic distribution gives greater aural prominence to the minimum value, while randomizations seem to extend upward from this minimum to the maximum.

The benefits of grain period are difficult to ascertain. Because the two share a common unit of measurement, namely milliseconds, the use of grain period instead of grains per second facilitates clearer comparisons with the grain duration. These comparisons in the new interface are obscured by the use of different scaling functions for their respective sliders. Some users may also find it unintuitive that the grains per second increases as they move the slider to the left, the direction typically associated with lower values on horizontal sliders. The benefits of mean and bandwidth constraints upon the grain period are less ambiguous. As the user increases and decreases the bandwidth, the mean value has a clear aural prominence when compared to the random deviations that surround it.

Potential differences between the specified constraints on randomization and the actual values produced by randomization were the catalyst for developing a feedback display, but the net result is an element that unifies the interface. The display provides a means for bridging the gap between the two control sliders by allowing users to select which scaling function is used to plot the period and duration values generated for each grain. Its placement between the two sliders further reinforces this role as an intermediary. Although the sound produced by an audio application should be its primary feedback mechanism, the usefulness of simultaneous visual feedback, such as provided by this display, should not be underestimated. Not only does it reinforce the information received by the ears, but visual feedback in close proximity with the visual controls facilitates quicker comparisons between input and output.

As the author stated in chapter 4, the experiments in this study should be viewed as the beginning of a larger research task aimed at understanding the perception of the granular sounds. Further experiments will be needed to make informed design changes involving the granular processing control parameters not studied here. Parameters such as sample offset, pitch multiplier, window shape, and amplitude would all make suitable independent variables for experiments designed to use MDS. An examination of more complicated granular processing textures, including examples with overlapping grains and multiple granular voices, will also be necessary in order to increase the external validity of this research. As this study showed, sound source should be avoided as an independent variable because of the tendency for these differences to become the primary focus of participants' responses. A logical follow-up to this study would be to repeat these experiments without the inclusion of sound source differences between stimuli in order to place greater emphasis on the processing differences.

Future studies must find another means of addressing the secondary inquiry into experience and its effects on perception. The definition of electroacoustic music given to participants was either inadequate or ambiguous, leading to large differences between their responses to the related questions.

Perhaps the secondary inquiry could have been more germane to the primary and involved questions about participants' familiarity and experience with granular techniques. Such an inquiry may be able to determine whether direct experience alters participants' perceptual organization of granular sounds. In order to improve the analysis of significant differences between subgroups, questions related to experience should also be used to develop subgroups of equal size prior to conducting the experiment sessions, something that was overlooked in the current study.

Future research into the perception of granular processing must also move beyond the use of MDS and explore other experiment designs and analysis methods. The preference for base-2 logarithmic scaling could be verified with an experiment in which participants listened to granular examples that have been generated using a variety of scaling factors and simply indicated their preferences. Designs that utilize semantic differential scales or verbal attributes (Kendall & Carterette, 1993a, 1993b) have proven effective in more recent timbre studies and may be a promising way to augment the findings of this study. Studies similar to the current one that are instead focused on related microsound techniques would provide a means of testing the how specific the findings are to granular processing. Because this study has proven the ability of empirical research to inform specific design issues in computer music software development, it will hopefully encourage other developers to embark on similar projects and report on their work.

REFERENCES

Apple Computer Inc. (2005). AudioUnit SDK overview [Computer software documentation]. Retrieved March 7, 2005, from ftp://ftp.apple.com/developer/Development_Kits/CoreAudioSDK1.3.4.dmg.bin

Audio Ease. (2002). *thOnk_0+2 – Free granular synthesis for Mac*. Retrieved May 11, 2004, from http://www.audioease.com/Pages/Free/FreeMain.html

Behles, G. (1998). Stampede II manual [Computer software documentation]. Retrieved December 31, 2003, from http://www.kgw.tu-berlin.de/~gb/stampede_manual.html

Behles, G., Starke, S., & Röbel, A. (1998). Quasi-synchronous and pitch-synchronous granular sound processing with Stampede II. *Computer Music Journal, 22 (2),* 44-51.

Bencina, R. (1998). *Oasis-Rose* the composition – Real-time DSP with AudioMulch. Paper presented at the 1998 Australasian Computer Music Conference. Retrieved December 17, 2003, from http://www.audiomulch.com/articles/OasisRose_ACMC98.pdf

Bencina, R. (2001). *Implementing real-time granular synthesis.* Unpublished manuscript. Retrieved December 17, 2003, from http://www.audiomulch.com/~rossb/rb-gst/BencinaAudioAnecdotes310801.pdf

Boulanger, R., (Ed.). (2000). *The Csound book: Perspectives in software synthesis, sound design, signal processing, and programming*. Cambridge, MA: MIT Press.

Butler, D. (1992). *The musician's guide to perception and cognition*. New York: Schirmer Books.

Ceruzzi, P. E. (1998.) *A history of modern computing.* Cambridge, MA: The MIT Press.

Chadabe, J. (1997). *Electric sound: The past and promise of electronic music.* Upper Saddle River, NJ: Prentice Hall.

Chowning, J. (1996). Foreword: New music and science. In C. Roads, *The computer music tutorial* (pp. ix-xii). Cambridge, MA: The MIT Press.

Cycling74. (2001). *Max 4: Getting started* [Computer software documentation]. Retrieved March 1, 2002, from http://www.synthesisters.com/download/ Max4GettingStarted.pdf

Cycling74. (2004). *Javascript in Max* {computer software documentation]. Retrieved August 1, 2004, from http://www.synthesisters.com/download/ JavaScriptInMax.pdf

DePoli, G. & Piccialli, A. (1991). Pitch-synchronous granular synthesis. In G. DePoli, A. Piccialli, & C. Roads (Eds.), *Representations of musical signals* (pp. 187-219). Cambridge, MA: MIT Press.

Dodge, C. and Jerse, T. A. (1997). *Computer music: Synthesis, composition, and performance* (2nd ed.). New York: Schirmer Books.

Eckel, G., Rocha-Iturbide, M. & Becker, B. (1995). The development of GiST, a granular synthesis toolkit based on an extension of the FOF generator. In R. Bidlack (Ed.), *Proceedings of the 1995 International Computer Music Conference* (pp. 296-302). San Francisco: International Computer Music Association.

Erbe, T. (2002). SoundHack (version 0.891) [Computer software]. Retrieved October 20, 2002, from http://www.soundhack.com/

Franke, N. (2002). SoundApp (version 2.6.1) [Computer software]. Retrieved January 20, 2002, from http://www.spies.com/~franke/SoundApp/

Gabor, D. (1947). Acoustical quanta and the theory of hearing. *Nature, 159 (4044),* 591-594.

Grey, J. M. (1977). Multidimensional perceptual scaling of musical timbres. *Journal of the Acoustical Society of America, 61 (5),* 1270-1277.

Helmholtz, H. L. F. (1912). *On the sensation of tone as a physiological basis for the theory of music* (4th ed.; A. J. Ellis, Trans.). London: Longmans, Green, and Co. (Original work published 1877)

Helmuth, M. (1991). Patchmix and StochGran: Two graphical interfaces. In B. Alphonce & B. Pennycook (Eds.), *Proceedings of the 1991 International Computer Music Conference* (pp. 563-566). San Francisco: International Computer Music Association.

Helmuth, M. (1993). Granular synthesis with Cmix and MAX. In S. Ohteru (Ed.), *Proceedings of the 1993 International Computer Music Conference* (pp. 449-452). San Francisco: International Computer Music Association.

Helmuth, M. & Ibrahim, A. (1995). The Fcurve sound generator with granular synthesis. In R. Bidlack (Ed.), *Proceedings of the 1995 International Computer Music Conference* (pp. 63-64). San Francisco: International Computer Music Association.

Hiller, L. (1989). Composing with computers: A progress report. In C. Roads (Ed.), *The music machine: Selected readings from Computer Music Journal* (pp. 75-89). Cambridge, MA: The MIT Press.

Iverson, P. & Krumhansl, C. L. (1993). Isolating the dynamic attributes of musical timbre. *Journal of the Acoustical Society of America, 94 (5),* 2595-2603.

Jones, D. L. & Parks, T. W. (1988). Generation and combination of grains of sound. *Computer Music Journal, 12 (2),* 27-33.

Keller, D. & Rolfe, C. (1998). *The corner effect.* Paper presented at the XII Colloquium on Musical Informatics, University of Udine, Gorizia, Italy. Retrieved December 15, 2003, from http://www.sfu.ca/~dkeller/CornerEffect/CornerEffect.html

Kendall, R. A. (2002). Music Experiment Development System – MEDS (version 2002-B1) [Computer software]. Retrieved October 20, 2002, from http://www.ethnomusic.ucla.edu/systematic/Faculty/Kendall/meds.htm

Kendall, R. A. & Carterette, E. C. (1991). Perceptual scaling of simultaneous wind instrument timbres. *Music Perception, 8 (4),* 369-404.

Kendall, R. A. & Carterette, E. C. (1993a). Verbal attributes of simultaneous wind instrument timbres: I. von Bismarck's adjectives. *Music Perception, 10 (4),* 445-468.

Kendall, R. A. & Carterette, E. C. (1993b). Verbal attributes of simultaneous wind instrument timbres: II. Adjectives induced from Piston's Orchestration. *Music Perception, 10 (4),* 469-502.

Kendall, R. A., Carterette, E. C., & Hajda, J. M. (1999). Perceptual and Acoustical Features of Natural and Synthetic Orchestral Tones. *Music Perception*, 16 (3), 327-364.

Kernigham, B. W. & Ritchie, D. M. (1988). *The C programming language* (2nd ed.). Englewood Cilffs, NJ: Prentice Hall.

Kruskal, J. B. (1964a). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika, 29 (1),* 1-27.

Kruskal, J. B. (1964b). Nonmetric multidimensional scaling: A numerical method. *Psychometrika, 29 (2),* 115-129.

Lansky, P. (1990). A view from the bus: When machines make music [Electronic version]. *Perspectives of New Music, 28 (2),* 102-110. Retrieved April 9, 2004, from http://silvertone.princeton.edu/~paul/view.html

Laske, O. E. (1988). Introduction to cognitive musicology. *Computer Music Journal, 12 (1),* 43-57.

Laske, O. E. (1989a). Composition theory in Koenig's Project One and Project Two. In C. Roads (Ed.), *The music machine: Selected readings from Computer Music Journal* (pp. 119-130). Cambridge, MA: The MIT Press.

Lakse, O. E. (1989b). Composition theory: An enrichment of music theory. *Interface, 18,* 45-59.

Lee, A. S. C. (1995). CSOUND granular synthesis unit generator. In R. Bidlack (Ed.), *Proceedings of the 1995 International Computer Music Conference* (pp. 230-231). San Francisco: International Computer Music Association.

Lee, A. S. C. (2000). Granular synthesis in Csound. In R. Boulanger (Ed.), *The Csound book: Perspectives in software synthesis, sound design, signal processing, and programming* (pp. 281-292). Cambridge, MA: MIT Press.

Lent, K. (1989). An efficient method for pitch shifting digitally sampled sounds. *Computer Music Journal, 13 (4),* 65-71.

Lindeman, E., Starkier, M., & Dechelle, F. (1990). The IRCAM Signal Processing Workstation: Hardware overview and signal processing features. In S. Arnold & G. Hair (Eds.), *Proceedings of the 1990 International Computer Music Conference* (pp. 132-135). San Francisco: International Computer Music Association.

Lippe, C. (1994). Real-time granular sampling using the IRCAM Signal Processing Workstation. *Contemporary Music Review, 10 (2),* 149-155.

Lyon, E. (2002). Dartmouth Symposium on the Future of Computer Music Software: A panel discussion. *Computer Music Journal, 26 (4),* 13-30.

McAdams, S. & Bregman, A. (1979). Hearing Musical Streams. *Computer Music Journal, 3 (4)*, 26-41.

McNabb, M. (1986). Computer music: Some aesthetic considerations. In S. Emmerson (Ed.), *The language of electroacoustic music* (pp. 141-153). New York: Harwood Academic Publishers.

Nelson, J. C. (2000a). GrainMaker 2 [Computer software]. In R. Boulanger (Ed.), *The Csound book: Perspectives in software synthesis, sound design, signal processing, and programming* [CD-ROM]. Cambridge, MA: MIT Press.

Nelson, J. C. (2000b). Granular Strategies. In R. Boulanger (Ed.), *The Csound book: Perspectives in software synthesis, sound design, signal processing, and programming* [CD-ROM]. Cambridge, MA: MIT Press.

Orton, R., Hunt, A., & Kirk, R. (1991). Graphical control of granular synthesis using cellular automata and the Freehand program. In B. Alphonce and B. Pennycook (Eds.), *Proceedings of the 1991 International Computer Music Conference* (pp. 416-418). San Francisco: International Computer Music Association.

Puckette, M. (1988). The patcher. In C. Lischka & J. Fritsch (Eds.), *Proceedings of the 1988 International Computer Music Conference* (pp. 420-429). San Francisco: International Computer Music Association.

Puckette, M. (1991). Combining event and signal processing in the Max graphical programming environment. *Computer Music Journal, 15 (3)*, 68-77.

Puckette, M. (2002). Max at seventeen. *Computer Music Journal*, 26 (4), 31-43.

Risset, J. C. & Wessel, D. L. (1999). Exploration of timbre by analysis and synthesis. In D. Deutsch (Ed.), *The psychology of music* (2nd ed.; pp. 113-169). San Diego, CA: Academic Press.

Roads, C. (1978). Automated granular synthesis of sound. *Computer Music Journal, 2 (2),* 61-62.

Roads, C. (1985). Granular synthesis of sound. In C. Roads & J. Strawn (Eds.), *Foundations of computer music* (pp. 146-159). Cambridge, MA: MIT Press.

Roads, C. (1991). Asynchronous granular synthesis. In G. DePoli, A, Piccialli, & C. Roads (Eds.), *Representations of musical signals* (pp. 143-186). Cambridge, MA: MIT Press.

Roads, C. (2001). *Microsound*. Cambridge, MA: MIT Press.

Roads, C. & Alexander, J. (1997). Cloud Generator [Computer software]. Retrieved September 1, 2002, from ftp://ftp.create.ucsb.edu/pub/CloudGenerator/cg.nofpu.hqx

Rolfe, C. & Keller, D. (2000). MacPod (version 1.3) [Computer software]. Retrieved August 28, 2002, from http://www.thirdmonk.com/Download/MacPod13.hqx

Shepard, R. N. (1962a). The analysis of proximities: Multidimensional scaling with an unknown distance function. I. *Psychometrika, 27 (2),* 125-140.

Shepard, R. N. (1962b). The analysis of proximities: Multidimensional scaling with an unknown distance function. II. *Psychometrika, 27 (3),* 219-246.

Steinberg Media Technologies GmbH. (1999). *Virtual studio technology plug-in specification 2.0: Software development kit* [Computer software documentation]. Retrieved March 7, 2005, from http://www.steinberg.net/steinberg/ygrabit/index.html

Tanghe, K. (2003). KTGranulator (version 1.1) [Computer software]. Retrieved December 17, 2003, from http://www.smartelectronix.com/users/koen/KTGranulator/

Todoroff, T. (1995). Real-time granular morphing and spatialisation of sounds with gestual control within Max/FTS. In R. Bidlack (Ed.), *Proceedings of the 1995 International Computer Music Conference* (pp. 315-318). San Francisco: International Computer Music Association.

Truax, B. (1986a). Computer music language design and the composing process. In S. Emmerson (Ed.), *The language of electroacoustic music* (pp. 155-173). New York: Harwood Academic Publishers.

Truax, B. (1986b). Real-time granular synthesis with the DMX-1000. In P. Berg (Ed.), *Proceedings of the 1986 International Computer Music Conference* (pp. 231-235). San Francisco: International Computer Music Association.

Truax, B. (1987). Real-time granulation of sampled sound with the DMX-1000. In S. Tipei and J. Beauchamp (Eds.), *Proceedings of the 1987 International Computer Music Conference* (pp. 138-145). San Francisco: International Computer Music Association.

Truax, B. (1988). Real-time granular synthesis with a digital signal processing computer. *Computer Music Journal, 12 (2),* 14-26.

Truax, B. (1990). Time-shifting of sampled sound with a real-time granulation technique. In S. Arnold & G. Hair (Eds.), *Proceedings of the 1990 International Computer Music Conference* (pp. 104-107). San Francisco: International Computer Music Association.

Truax, B. (1994). Discovering inner complexity: Time shifting and transposition with real-time granulation technique. *Computer Music Journal, 18 (2),* 38-48.

van der Schoot, A. (1999). thOnk 0+2 [Computer software]. Retrieved September 5, 2002, from http://www.audioease.com/download/thOnk_0+2.sit.hqx

Vercoe, B. L. (1997). The Csound manual: A manual for the audio-processing system (version 3.47) [Computer software documentation]. Jean Piché, Ed. Retrieved March 8, 2005, from http://www.ugrad.physics.mcgill.ca/reference/Csound/TITLE.html (Original work published 1986)

Wessel, D. L. (1979). Timbre space as a musical control structure. *Computer Music Journal , 3 (2),* 45-52.

Wolek, N. (2001). Granular Toolkit v1.0 for Cycling74's Max/MSP. *Journal SEAMUS, XVI (2)*, 34-46.

Xenakis, I. (1992). *Formalized music: Thought and mathematics in music* (Rev. ed.). Stuvesant, NY: Pendragon Press. (Original work published in 1963)

Zicarelli, D. (2002). How I learned to love a program that does nothing. *Computer Music Journal, 26 (4),* 44-51.

APPENDIX A: RAW COORDINATES FROM MDS SOLUTIONS

Table 23. Coordinates for the Two- and Three-Dimensional MDS Solutions from Experiment 1

	2-D Solution [a]		3-D Solution [b]		
Stimuli Reference	1	2	1	2	3
b_p57_l15	-1.0257	-1.5496	-1.2117	-1.8621	-0.3043
b_p57_l22	-1.1913	-0.8899	-1.4207	-1.0299	0.4597
b_p57_l29	-1.3714	-0.2728	-1.5275	-0.2751	0.8593
b_p75_l22	-1.2241	-0.093	-1.4453	-0.1292	-0.3491
b_p75_l29	-1.0285	0.495	-1.2347	0.5813	0.2664
b_p75_l36	-0.9368	0.6883	-1.1192	0.8075	0.3429
b_p93_l29	-0.8772	0.8009	-0.9666	0.8852	-0.7131
b_p93_l36	-0.7408	1.0775	-0.8405	1.2408	-0.5721
b_p93_l43	-0.8282	1.278	-0.9673	1.5199	-0.3845
v_p57_l15	0.3717	-2.0708	0.4638	-2.2823	-1.0994
v_p57_l22	0.7887	-1.2114	0.9336	-1.4604	0.2185
v_p57_l29	1.0575	-0.6841	1.0402	-0.6919	1.1025
v_p75_l22	1.0047	-0.5839	1.2009	-0.7325	-0.1376
v_p75_l29	1.1685	0.2398	1.3748	0.2522	0.3732
v_p75_l36	1.2112	0.5725	1.3702	0.6308	0.6576
v_p93_l29	1.2312	0.4617	1.4689	0.5101	-0.3633
v_p93_l36	1.1646	0.9156	1.4103	1.0795	-0.0163
v_p93_l43	1.2259	0.8261	1.4707	0.9563	-0.3404

Note. Stimuli References are comprised of the first letter of the sound source (b = bell, v = voice), program setting for grain period and program setting for grain length.

[a] stress = 0.13587, R^2 = 0.90542. [b] stress = 0.08832, R^2 = 0.94444.

Table 24. Coordinates for the Two- and Three-Dimensional MDS Solutions from Experiment 2

	2-D Solution [a]		3-D Solution [b]		
Stimuli Reference	1	2	1	2	3
b_l22_lbw0	1.2881	0.4327	1.3792	0.3229	-1.0506
b_l22_lbw125	1.1272	1.156	1.3758	1.3623	0.148
b_l22_lbw160	1.0009	1.6241	1.2299	1.924	0.2065
b_l29_lbw0	1.2354	-0.4072	1.4801	-0.5016	0.0179
b_l29_lbw100	1.1809	-0.0621	1.4169	-0.0902	-0.1004
b_l29_lbw125	1.2498	-0.0473	1.4919	-0.0428	0.283
b_l36_lbw0	1.2651	-0.8145	1.4912	-0.9551	-0.4431
b_l36_lbw50	1.2512	-0.681	1.5021	-0.8046	0.2007
b_l36_lbw100	1.3087	-0.5106	1.542	-0.5571	0.5045
v_l22_lbw0	-1.093	-0.3092	-1.1633	-0.2416	-1.0431
v_l22_lbw125	-1.4207	0.7715	-1.7159	0.9009	-0.2509
v_l22_lbw160	-1.3352	1.0314	-1.6055	1.2285	0.2349
v_l29_lbw0	-1.0352	-0.6142	-1.2597	-0.7232	-0.23
v_l29_lbw100	-1.4712	-0.102	-1.7507	-0.1181	0.4005
v_l29_lbw125	-1.342	0.255	-1.6195	0.2785	-0.2347
v_l36_lbw0	-0.851	-0.8568	-1.0307	-0.9935	0.3597
v_l36_lbw50	-1.0824	-0.6697	-1.3081	-0.7814	0.2877
v_l36_lbw100	-1.2765	-0.1959	-1.4558	-0.2079	0.7094

Note. Stimuli References are comprised of the first letter of the sound source (b = bell, v = voice), program setting for grain length (ms) and program setting for length bandwidth (%).

[a] stress = 0.13045, R^2 = 0.92762. [b] stress = 0.08268, R^2 = 0.96403.

Table 25. Coordinates for the Two- and Three-Dimensional MDS Solutions from Experiment 3 (Non-listeners, n = 5)

	2-D Solution [a]		3-D Solution [b]		
Stimuli Reference	1	2	1	2	3
b_p57_pbw0	1.1972	-0.8243	1.4045	-1.0237	0.0686
b_p57_pbw50	1.0989	-0.7152	1.2809	-0.9036	0.0361
b_p57_pbw100	1.1055	-0.5353	1.3009	-0.626	0.2253
b_p75_pbw0	1.2952	0.2994	1.3941	-0.0882	-0.9553
b_p75_pbw100	1.0632	0.3658	1.2636	0.4614	-0.0155
b_p75_pbw125	1.0456	0.2967	1.2228	0.4566	0.2788
b_p93_pbw0	1.5255	0.2271	1.8176	0.2108	-0.2178
b_p93_pbw125	1.469	0.3378	1.7016	0.5603	0.4543
b_p93_pbw160	1.385	0.3914	1.5991	0.6569	0.4395
v_p57_pbw0	-1.1548	-0.7642	-1.3675	-0.9776	-0.3686
v_p57_pbw50	-1.4476	-0.3928	-1.7114	-0.5963	-0.3098
v_p57_pbw100	-1.1182	-0.4883	-1.3322	-0.3193	0.586
v_p75_pbw0	-1.3067	-0.3619	-1.5303	-0.5477	-0.4485
v_p75_pbw100	-1.4426	0.1424	-1.6718	0.432	0.4581
v_p75_pbw125	-1.0419	-0.3904	-1.2636	-0.2651	0.3708
v_p93_pbw0	-0.735	1.9351	-0.8524	1.2814	-1.9774
v_p93_pbw125	-1.5055	-0.0562	-1.6297	0.4498	0.9133
v_p93_pbw160	-1.4327	0.5325	-1.6262	0.8385	0.4619

Note. Stimuli References are comprised of the first letter of the sound source (b = bell, v = voice), program setting for grain period (ms) and program setting for period bandwidth (%).

[a] stress = 0.16865, R^2 = 0.88988. [b] stress = 0.16211, R^2 = 0.90549.

Table 26. Coordinates for the Two- and Three-Dimensional MDS Solutions from Experiment 3 (Listeners, n = 17)

	2-D Solution [a]		3-D Solution [b]		
Stimuli Reference	1	2	1	2	3
b_p57_pbw0	1.1063	-0.9039	1.2385	0.8869	0.9449
b_p57_pbw50	1.2108	-0.8319	1.4438	1.009	0.1754
b_p57_pbw100	1.2051	-0.4942	1.4313	0.5952	-0.2736
b_p75_pbw0	1.2894	0.167	1.5158	-0.1777	0.4188
b_p75_pbw100	1.177	0.3857	1.3857	-0.4204	-0.4336
b_p75_pbw125	1.2703	0.0043	1.5028	0.0122	-0.3582
b_p93_pbw0	1.1955	1.0041	1.4169	-1.1554	0.4541
b_p93_pbw125	1.2322	0.2394	1.4001	-0.2323	-0.6991
b_p93_pbw160	1.4684	0.1244	1.7515	-0.1362	-0.2963
v_p57_pbw0	-1.2659	-0.6683	-1.4818	0.6874	0.6714
v_p57_pbw50	-1.2725	-0.6915	-1.519	0.7736	0.4424
v_p57_pbw100	-1.2869	-0.3993	-1.5218	0.4315	0.4732
v_p75_pbw0	-1.2698	-0.0329	-1.5262	0.0259	-0.1974
v_p75_pbw100	-1.2655	-0.2835	-1.4528	0.2848	-0.6786
v_p75_pbw125	-1.333	-0.1248	-1.6007	0.143	-0.1999
v_p93_pbw0	-0.8988	1.8962	-1.0575	-2.0877	0.9965
v_p93_pbw125	-1.2046	0.576	-1.403	-0.6074	-0.629
v_p93_pbw160	-1.3583	0.0334	-1.5236	-0.0324	-0.8111

Note. Stimuli References are comprised of the first letter of the sound source (b = bell, v = voice), program setting for grain period (ms) and program setting for period bandwidth (%).

[a] stress = 0.12496, R^2 = 0.95607. [b] stress = 0.07412, R^2 = 0.97158.

Table 27. Coordinates for the Two- and Three-Dimensional MDS Solutions from Experiment 3 (Non-composers, n = 9)

	2-D Solution [a]		3-D Solution [b]		
Stimuli Reference	1	2	1	2	3
b_p57_pbw0	1.2734	0.591	1.4653	0.6089	0.685
b_p57_pbw50	1.2249	0.7819	1.4525	0.8997	0.3859
b_p57_pbw100	1.1065	0.6712	1.3212	0.8058	-0.0825
b_p75_pbw0	1.2763	-0.2482	1.4754	-0.2608	0.5678
b_p75_pbw100	1.0889	-0.4764	1.2933	-0.5237	-0.36
b_p75_pbw125	1.1431	0.0029	1.3389	0.0216	-0.4313
b_p93_pbw0	1.3364	-0.7683	1.5959	-0.8816	0.3096
b_p93_pbw125	1.2882	-0.2275	1.4736	-0.2098	-0.6777
b_p93_pbw160	1.4671	-0.0897	1.6786	-0.0753	-0.7256
v_p57_pbw0	-1.2007	0.6684	-1.3997	0.6494	0.724
v_p57_pbw50	-1.3206	0.5772	-1.567	0.6181	0.4982
v_p57_pbw100	-1.2738	0.4244	-1.5415	0.4841	0.0153
v_p75_pbw0	-1.2411	0.2074	-1.4558	0.1939	0.498
v_p75_pbw100	-1.3582	0.0825	-1.5716	0.0679	-0.6283
v_p75_pbw125	-1.133	0.3469	-1.367	0.3913	-0.1382
v_p93_pbw0	-0.8127	-2.0023	-0.9488	-2.2388	0.9359
v_p93_pbw125	-1.3862	-0.4356	-1.61	-0.4645	-0.6376
v_p93_pbw160	-1.4784	-0.1056	-1.6332	-0.0862	-0.9385

Note. Stimuli References are comprised of the first letter of the sound source (b = bell, v = voice), program setting for grain period (ms) and program setting for period bandwidth (%).

[a] stress = 0.12884, R^2 = 0.93920. [b] stress = 0.10989, R^2 = 0.94926.

Table 28. Coordinates for the Two- and Three-Dimensional MDS Solutions from Experiment 3 (Composers, n = 13)

	2-D Solution [a]		3-D Solution [b]		
Stimuli Reference	1	2	1	2	3
b_p57_pbw0	1.0272	-1.0225	1.1615	1.1291	0.7876
b_p57_pbw50	1.1377	-0.8844	1.3539	1.0695	-0.1635
b_p57_pbw100	1.2129	-0.4421	1.4407	0.5173	-0.2655
b_p75_pbw0	1.302	0.1368	1.5167	-0.1009	0.5019
b_p75_pbw100	1.2016	0.2295	1.4142	-0.2965	-0.3384
b_p75_pbw125	1.2851	0.0776	1.5067	-0.1197	-0.427
b_p93_pbw0	1.236	0.9639	1.4544	-1.0118	0.6726
b_p93_pbw125	1.2918	0.2801	1.4693	-0.3458	-0.6637
b_p93_pbw160	1.4494	0.1624	1.7255	-0.2126	-0.2799
v_p57_pbw0	-1.2797	-0.6925	-1.4877	0.7525	0.6693
v_p57_pbw50	-1.3151	-0.6722	-1.5751	0.7941	0.2936
v_p57_pbw100	-1.2266	-0.4117	-1.4583	0.4737	0.3841
v_p75_pbw0	-1.3174	-0.0143	-1.5896	0.0153	0.0848
v_p75_pbw100	-1.2698	-0.3243	-1.431	0.2126	-0.8426
v_p75_pbw125	-1.3718	0.0001	-1.6508	-0.0373	-0.1396
v_p93_pbw0	-0.8773	1.8365	-1.0167	-1.922	1.1709
v_p93_pbw125	-1.1767	0.6178	-1.3542	-0.6824	-0.6879
v_p93_pbw160	-1.3096	0.1591	-1.4795	-0.235	-0.7566

Note. Stimuli References are comprised of the first letter of the sound source (b = bell, v = voice), program setting for grain period (ms) and program setting for period bandwidth (%).

[a] stress = 0.13607, R^2 = 0.94285. [b] stress = 0.08519, R^2 = 0.96148.

APPENDIX B: JAVASCRIPT SOURCE FOR NEW UI ELEMENTS

A. JavaScript code for slider to control grain duration

```
/*

jsui_maxminslider.js
min and max slider, controlled by x & y mouse movements
by Nathan Wolek

arguments: handle(RGB) background(RGB) bandwidth(RGB) track(RGB)

*/
/*******************************************/
// setup object
inlets = 2;
outlets = 2;
sketch.default2d();
// colors
var vbrgb = [1.,1.,1.];          // background
var vfrgb = [1.,0.2,0.2]; // handle
var vrgb2 = [1.,0.5,0.5]; // bandwidth box
var vrgb3 = [0,0,0]; // track line
// drawing
var alpha_level = 0.9;
var handle_size = 0.5;
var roundness = 0.3;
var bw_height = handle_size * 0.5;
var voutline = 0;
// var handling
var v_min = 4.;
var v_max = 7.5;
var v_diff = v_max - v_min;
// variables
var min_val = {ms:0,val:0,x_val:0};
var max_val = {ms:0,val:0,x_val:0};
//  mouse movement tracking
var vx = 0.;
var last_vx = 0.;
var vy = 0.;
```

```
var last_vy = 0.;
var firstdown = 0;
var track = 0;

set_min(5.);
set_max(6.);

if (jsarguments.length>1)
	vfrgb[0] = jsarguments[1]/255.;
if (jsarguments.length>2)
	vfrgb[1] = jsarguments[2]/255.;
if (jsarguments.length>3)
	vfrgb[2] = jsarguments[3]/255.;
if (jsarguments.length>4)
	vbrgb[0] = jsarguments[4]/255.;
if (jsarguments.length>5)
	vbrgb[1] = jsarguments[5]/255.;
if (jsarguments.length>6)
	vbrgb[2] = jsarguments[6]/255.;
if (jsarguments.length>7)
	vrgb2[0] = jsarguments[7]/255.;
if (jsarguments.length>8)
	vrgb2[1] = jsarguments[8]/255.;
if (jsarguments.length>9)
	vrgb2[2] = jsarguments[9]/255.;
if (jsarguments.length>10)
	vrgb3[0] = jsarguments[10]/255.;
if (jsarguments.length>11)
	vrgb3[1] = jsarguments[11]/255.;
if (jsarguments.length>12)
	vrgb3[2] = jsarguments[12]/255.;

draw();
refresh();

function draw()
{
	var width = box.rect[2] - box.rect[0];
	var height = box.rect[3] - box.rect[1];
	var aspect = width/height;

	var mean = (max_val.x_val+min_val.x_val)*0.5;
```

```
		var drawing_width = (max_val.x_val-min_val.x_val)*0.5;
		//post("mean: " + mean + ", bw: " + bw + "\n");
		//DEBUG MSG

		with (sketch) {
			//scale everything to box size
			glmatrixmode("modelview");
			glpushmatrix();
			glscale(aspect,1,1);
			glenable("line_smooth");

			// erase background
			glclearcolor(vbrgb);
			glclear();

			//draw track line
			glcolor(vrgb3,1);
			shapeslice(1,1);
			linesegment(-1,0,0,1,0,0);

			// draw bandwidth
			moveto(mean,0);
			if (voutline) {
				glcolor(0,0,0,1);
				glpolygonmode("front_and_back","line");
				plane(drawing_width,bw_height);
				glpolygonmode("front_and_back","fill");
			}
			glcolor(vrgb2, alpha_level);
			plane(drawing_width,bw_height);

			// draw handle
			moveto(min_val.x_val,0);
			if (voutline) {
				glcolor(0,0,0,1);
				glpolygonmode("front_and_back","line");
				roundedplane(roundness,(handle_size /
aspect),handle_size);
				glpolygonmode("front_and_back","fill");
				glcolor(1,1,1,1);
				roundedplane(roundness,(handle_size /
aspect),handle_size);
```

```
		}
		glcolor(vbrgb, alpha_level);
		roundedplane(roundness,(handle_size / aspect),handle_size);
		glcolor(vfrgb, alpha_level);
		roundedplane(roundness,(handle_size / aspect),handle_size);


		//reset transformation matrix
		glmatrixmode("modelview");
		glpopmatrix();
	}
}

function bang()
{
	draw();
	refresh();
	outlet(0,min_val.ms);
	outlet(1,max_val.ms);
}

function msg_float(v)
{
	switch(inlet)
	{
		case 0:
			set_min(v);
			break;
		case 1:
			set_max(v);
			break;
	}
	draw();
	refresh();
}

function fsaa(v)
{
	sketch.fsaa = v;
	bang();
}
```

```
function frgb(r,g,b)
{
	vfrgb[0] = r/255.;
	vfrgb[1] = g/255.;
	vfrgb[2] = b/255.;
	draw();
	refresh();
}

function rgb2(r,g,b)
{
	vrgb2[0] = r/255.;
	vrgb2[1] = g/255.;
	vrgb2[2] = b/255.;
	draw();
	refresh();
}

function brgb(r,g,b)
{
	vbrgb[0] = r/255.;
	vbrgb[1] = g/255.;
	vbrgb[2] = b/255.;
	draw();
	refresh();
}

function outline(v)
{
	voutline = v;
	draw();
	refresh();
}

function set_min(v)
{
	var nv = constrain(v,v_min,v_max);

	min_val.ms = nv;
	min_val.val = (min_val.ms-v_min)/v_diff;
	min_val.x_val = 1.8*min_val.val-0.9;
```

```
        //outlet(0,min_val.ms);
}
set_min.local = 1; //private.

function set_min_xval(v)
{
        var nv = constrain(v,-0.9,0.9);

        min_val.x_val = nv;
        min_val.val = (min_val.x_val+0.9)/1.8;
        min_val.ms = (min_val.val*v_diff)+v_min;

        //min_val.ms = Math.round(min_val.ms);

        //outlet(0,min_val.ms);
}
set_min_xval.local = 1; //private.

function set_max(v)
{
        var nv = constrain(v,min_val.ms,v_max);

        max_val.ms = nv;
        max_val.val = (max_val.ms-v_min)/v_diff;
        max_val.x_val = 1.8*max_val.val-0.9;

        //post(nv + " " + max_val.ms + " " + max_val.val + " " + max_val.x_val + "
\n"); //DEBUG MSG
        //outlet(1,max_val.ms);
}
set_max.local = 1; //private.

function set_max_xval(v)
{
        var nv = constrain(v,min_val.x_val,0.9);

        max_val.x_val = nv;
        max_val.val = (max_val.x_val+0.9)/1.8;
        max_val.ms = (max_val.val*v_diff)+v_min;

        //max_val.ms = Math.round(max_val.ms);
```

```
	//post(nv + " " + max_val.ms + " " + max_val.val + " " + max_val.x_val + "
\n"); //DEBUG MSG
	//outlet(1,max_val.ms);
}
set_max_xval.local = 1; //private.

function constrain(v,min,max)
{
	var nv = Math.min(Math.max(min,v),max);
	return nv;
}
constrain.local = 1; //private.

//
// User Interaction
//

function eval_mouse(v)
{
	var tx,dx,ty,dy;
	var width = box.rect[2] - box.rect[0];
	var height = box.rect[3] - box.rect[1];
	var aspect = width/height;

	vx = v[0]/aspect;
	vy = v[1]/aspect;

	if(firstdown)
	{
		if(Math.abs(vx-min_val.x_val)<0.1)
		{
			track = 1;
			//post("handle is being tracked!!!\n");
	// DEBUG MSG
		}
		firstdown = 0;
		last_vx = vx;
		last_vy = vy;
	}

	if (track)
	{
```

```
		dx = vx - last_vx;
		tx = min_val.x_val + dx;
		set_min_xval(tx);

		dy = vy - last_vy;
		ty = max_val.x_val + dy;
		set_max_xval(ty);

	}

	last_vx = vx;
	last_vy = vy;

}
eval_mouse.local = 1; //private.

function onclick(x,y,but,cmd,shift,capslock,option,ctrl)
{
	firstdown = 1;
	track = 0;
}
onclick.local = 1; //private. could be left public to permit "synthetic" events

function ondrag(x,y,but,cmd,shift,capslock,option,ctrl)
{
	var w = sketch.screentoworld(x,y);
	eval_mouse(w);
	bang();
}
ondrag.local = 1; //private. could be left public to permit "synthetic" events

function onresize(w,h)
{
	// erase background
	sketch.glclearcolor(vbrgb);
	sketch.glclear();
	draw();
	refresh();
}
onresize.local = 1; //private
/*******************************************/
```

B. JavaScript code for slider to control grain period.

```
/*

jsui_meanbwslider.js
Mean and bandwidth slider
by Nathan Wolek

arguments: handle(RGB) background(RGB) bandwidth(RGB) track(RGB)

*/
/*******************************************/
// setup object
inlets = 2;
outlets = 2;
sketch.default2d();
// colors
var vbrgb = [1.,1.,1.];          // background
var vfrgb = [0.2,0.2,1.]; // handle
var vrgb2 = [0.5,0.5,1.]; // bandwidth box
var vrgb3 = [0,0,0]; // track line
// drawing
var alpha_level = 0.85;
var handle_size = 0.5;
var roundness = 0.3;
var bw_height = handle_size * 0.5;
var voutline = 0;
// var handling
var mean_min = 20;
var mean_max = 250;
var mean_diff = mean_max - mean_min;
var bw_min = 1;
var bw_max = 250;
var bw_diff = bw_max - bw_min;
// variables
var mean = {ms:0,val:0,x_val:0};
var bandwidth = {ms:0,val:0,x_val:0};
//  mouse movement tracking
var vx = 0.;
var last_vx = 0.;
var vy = 0.;
var last_vy = 0.;
```

```
var firstdown = 0;
var track = 0;

set_mean(200.);
set_bandwidth(20.);

if (jsarguments.length>1)
	vfrgb[0] = jsarguments[1]/255.;
if (jsarguments.length>2)
	vfrgb[1] = jsarguments[2]/255.;
if (jsarguments.length>3)
	vfrgb[2] = jsarguments[3]/255.;
if (jsarguments.length>4)
	vbrgb[0] = jsarguments[4]/255.;
if (jsarguments.length>5)
	vbrgb[1] = jsarguments[5]/255.;
if (jsarguments.length>6)
	vbrgb[2] = jsarguments[6]/255.;
if (jsarguments.length>7)
	vrgb2[0] = jsarguments[7]/255.;
if (jsarguments.length>8)
	vrgb2[1] = jsarguments[8]/255.;
if (jsarguments.length>9)
	vrgb2[2] = jsarguments[9]/255.;
if (jsarguments.length>10)
	vrgb3[0] = jsarguments[10]/255.;
if (jsarguments.length>11)
	vrgb3[1] = jsarguments[11]/255.;
if (jsarguments.length>12)
	vrgb3[2] = jsarguments[12]/255.;

draw();
refresh();

function draw()
{
	var width = box.rect[2] - box.rect[0];
	var height = box.rect[3] - box.rect[1];
	var aspect = width/height;

	with (sketch) {
		//scale everything to box size
```

```
		glmatrixmode("modelview");
		glpushmatrix();
		glscale(aspect,1,1);
		glenable("line_smooth");

		// erase background
		glclearcolor(vbrgb);
		glclear();

		//draw track line
		glcolor(vrgb3,1);
		shapeslice(1,1);
		linesegment(-1,0,0,1,0,0);

		// draw bandwidth
		moveto(mean.x_val,0);
		if (voutline) {
			glcolor(0,0,0,1);
			glpolygonmode("front_and_back","line");
			plane(bandwidth.x_val,bw_height);
			glpolygonmode("front_and_back","fill");
		}
		glcolor(vrgb2, alpha_level);
		plane(bandwidth.x_val,bw_height);

		// draw handle
		moveto(mean.x_val,0);
		if (voutline) {
			glcolor(0,0,0,1);
			glpolygonmode("front_and_back","line");
			roundedplane(roundness,(handle_size /
aspect),handle_size);
			glpolygonmode("front_and_back","fill");
			glcolor(1,1,1,1);
			roundedplane(roundness,(handle_size /
aspect),handle_size);
		}
		glcolor(vfrgb, alpha_level);
		roundedplane(roundness,(handle_size / aspect),handle_size);


		//reset transformation matrix
```

```
		glmatrixmode("modelview");
		glpopmatrix();
	}
}

function bang()
{
	draw();
	refresh();
	outlet(0,mean.ms);
	outlet(1,bandwidth.ms);
}

function msg_float(v)
{
	switch(inlet)
	{
		case 0:
			set_mean(v);
			break;
		case 1:
			set_bandwidth(v);
			break;
	}
	draw();
	refresh();
}

function fsaa(v)
{
	sketch.fsaa = v;
	bang();
}

function frgb(r,g,b)
{
	vfrgb[0] = r/255.;
	vfrgb[1] = g/255.;
	vfrgb[2] = b/255.;
	draw();
	refresh();
}
```

```
function rgb2(r,g,b)
{
	vrgb2[0] = r/255.;
	vrgb2[1] = g/255.;
	vrgb2[2] = b/255.;
	draw();
	refresh();
}

function brgb(r,g,b)
{
	vbrgb[0] = r/255.;
	vbrgb[1] = g/255.;
	vbrgb[2] = b/255.;
	draw();
	refresh();
}

function outline(v)
{
	voutline = v;
	draw();
	refresh();
}

function set_mean(v)
{
	var nv = constrain(v,mean_min,mean_max);

	mean.ms = nv;
	mean.val = (mean.ms-mean_min)/mean_diff;
	mean.x_val = 1.8*mean.val-0.9;

	//outlet(0,mean.ms);
}
set_mean.local = 1; //private.

function set_mean_xval(v)
{
	var nv = constrain(v,-0.9,0.9);
```

```
	mean.x_val = nv;
	mean.val = (mean.x_val+0.9)/1.8;
	mean.ms = (mean.val*mean_diff)+mean_min;

	mean.ms = Math.round(mean.ms);

	//outlet(0,mean.ms);
}
set_mean_xval.local = 1; //private.

function set_bandwidth(v)
{
	var nv = constrain(v,bw_min,bw_max);

	bandwidth.ms = nv;
	bandwidth.val = (bandwidth.ms-bw_min)/bw_diff;
	bandwidth.x_val = 0.9*bandwidth.val;

	//post(nv + " " + bandwidth.ms + " " + bandwidth.val + " " +
bandwidth.x_val + " \n"); //DEBUG MSG
	//outlet(1,bandwidth.ms);
}
set_bandwidth.local = 1; //private.

function set_bandwidth_xval(v)
{
	var nv = constrain(v,0.,0.9);

	bandwidth.x_val = nv;
	bandwidth.val = (bandwidth.x_val)/0.9;
	bandwidth.ms = (bandwidth.val*bw_diff)+bw_min;

	bandwidth.ms = Math.round(bandwidth.ms);

	//post(nv + " " + bandwidth.ms + " " + bandwidth.val + " " +
bandwidth.x_val + " \n"); //DEBUG MSG
	//outlet(1,bandwidth.ms);
}
set_bandwidth_xval.local = 1; //private.

function constrain(v,min,max)
{
```

```
	var nv = Math.min(Math.max(min,v),max);
	return nv;
}
constrain.local = 1; //private.

//
// User Interaction
//

function eval_mouse(v)
{
	var tx,dx,ty,dy;
	var width = box.rect[2] - box.rect[0];
	var height = box.rect[3] - box.rect[1];
	var aspect = width/height;

	vx = v[0]/aspect;
	vy = v[1]/aspect;

	if(firstdown)
	{
		if(Math.abs(vx-mean.x_val)<0.1)
		{
			track = 1;
			//post("handle is being tracked!!!\n");
	// DEBUG MSG
		}
		firstdown = 0;
		last_vx = vx;
		last_vy = vy;
	}

	if (track)
	{
		dx = vx - last_vx;
		tx = mean.x_val + dx;
		set_mean_xval(tx);

		dy = vy - last_vy;
		ty = bandwidth.x_val + dy;
		set_bandwidth_xval(ty);
```

```
        }

        last_vx = vx;
        last_vy = vy;

}
eval_mouse.local = 1; //private.

function onclick(x,y,but,cmd,shift,capslock,option,ctrl)
{
        firstdown = 1;
        track = 0;
}
onclick.local = 1; //private. could be left public to permit "synthetic" events

function ondrag(x,y,but,cmd,shift,capslock,option,ctrl)
{
        var w = sketch.screentoworld(x,y);
        eval_mouse(w);
        bang();
}
ondrag.local = 1; //private. could be left public to permit "synthetic" events

function onresize(w,h)
{
        // erase background
        sketch.glclearcolor(vbrgb);
        sketch.glclear();
        draw();
        refresh();
}
onresize.local = 1; //private
/*******************************************/
```

C. JavaScript code for display of grain duration and period values.

```
/*

jsui_durperdisplay.js
length and period display
by Nathan Wolek
```

arguments: background(RGB) period_dots(RGB) duration_dots(RGB)

```
*/
/*******************************************/
// setup object
inlets = 1;
setinletassist(-1,in_describe);
sketch.default2d();
// colors
var vbrgb = [1.,1.,1.];		// background
var vfrgb = [0.05,0.05,1.]; // period dots
var vrgb2 = [1.,0.05,0.05]; // duration dots
// drawing
var alpha_level = 0.9;
var dot_size = 0.1;
var voutline = 0;
// scaling
var scale = 0; // scaling factor for sliders, 0 = linear, 1 = log2
var min_val_lin = 20;
var max_val_lin = 250;
var diff_val_lin = max_val_lin - min_val_lin; // DO NOT CHANGE
var log2 = Math.log(2.); // log 2 base e, used for log computations, DO NOT
CHANGE
var min_val_log2 = 4.;
var max_val_log2 = 7.5;
var diff_val_log2 = max_val_log2 - min_val_log2; // DO NOT CHANGE
// variables
var scale = 0; // scaling factor for sliders, 0 = linear, 1 = log2
var per_vals = new Array(-2,-2,-2,-2,-2,-2,-2,-2,-2,-2);
var per_log2 = new Array(-2,-2,-2,-2,-2,-2,-2,-2,-2,-2);
var per_index = per_vals.length;
var dur_vals = new Array(-2,-2,-2,-2,-2,-2,-2,-2,-2,-2);
var dur_log2 = new Array(-2,-2,-2,-2,-2,-2,-2,-2,-2,-2);
var dur_index = dur_vals.length;

if (jsarguments.length>1)
	vbrgb[0] = jsarguments[1]/255.;
if (jsarguments.length>2)
	vbrgb[1] = jsarguments[2]/255.;
if (jsarguments.length>3)
	vbrgb[2] = jsarguments[3]/255.;
if (jsarguments.length>4)
```

```
        vfrgb[0] = jsarguments[4]/255.;
if (jsarguments.length>5)
        vfrgb[1] = jsarguments[5]/255.;
if (jsarguments.length>6)
        vfrgb[2] = jsarguments[6]/255.;
if (jsarguments.length>7)
        vrgb2[0] = jsarguments[7]/255.;
if (jsarguments.length>8)
        vrgb2[1] = jsarguments[8]/255.;
if (jsarguments.length>9)
        vrgb2[2] = jsarguments[9]/255.;
if (jsarguments.length>10)
        vrgb3[0] = jsarguments[10]/255.;
if (jsarguments.length>11)
        vrgb3[1] = jsarguments[11]/255.;
if (jsarguments.length>12)
        vrgb3[2] = jsarguments[12]/255.;

bang();

function draw()
{
        var width = box.rect[2] - box.rect[0];
        var height = box.rect[3] - box.rect[1];
        var aspect = width/height;

        var dur_count = dur_index;
        var per_count = per_index;
        var temp, scaled_val;

        with (sketch) {
                //scale everything to box size
                glmatrixmode("modelview");
                glpushmatrix();
                //glscale(aspect,1,1);
                glenable("line_smooth");

                // erase background
                glclearcolor(vbrgb);
                glclear();

                // draw dur dots
```

```
shapeslice(5,5);
glcolor(vrgb2, alpha_level);
for (var i=0; i<dur_vals.length; i++)
{

	dur_count -= 1;
	while (dur_count < 0) dur_count += dur_vals.length;

	if (scale)
	{
		scaled_val = dur_log2[dur_count];
	} else {
		scaled_val = dur_vals[dur_count];
	}
	scaled_val = scaled_val * aspect;

	temp = 0.9 + ((i/dur_vals.length)*-1.8);
	moveto(scaled_val,temp);
	sphere(dot_size);

}

// draw per dots
glcolor(vfrgb, alpha_level);
for (var i=0; i<per_vals.length; i++)
{

	per_count -= 1;
	while (per_count < 0) per_count += per_vals.length;

	if (scale)
	{
		scaled_val = per_log2[per_count];
	} else {
		scaled_val = per_vals[per_count];
	}
	scaled_val = scaled_val * aspect;

	temp = 0.9 + ((i/per_vals.length)*-1.8);
	moveto(scaled_val,temp);
	sphere(dot_size);
```

```
		}

		//reset transformation matrix
		glmatrixmode("modelview");
		glpopmatrix();
	}
}

function bang()
{
	draw();
	refresh();
}

function dur(v)
{
	set_next_dur(v);
	bang();
}

function per(v)
{
	set_next_per(v);
}

function fsaa(v)
{
	sketch.fsaa = v;
	bang();
}

function frgb(r,g,b)
{
	vfrgb[0] = r/255.;
	vfrgb[1] = g/255.;
	vfrgb[2] = b/255.;
	draw();
	refresh();
}

function rgb2(r,g,b)
{
```

```
        vrgb2[0] = r/255.;
        vrgb2[1] = g/255.;
        vrgb2[2] = b/255.;
        draw();
        refresh();
}

function brgb(r,g,b)
{
        vbrgb[0] = r/255.;
        vbrgb[1] = g/255.;
        vbrgb[2] = b/255.;
        draw();
        refresh();
}

function outline(v)
{
        voutline = v;
        draw();
        refresh();
}

function set_next_dur(v)
{
        var temp, temp_l2;

        dur_index -= 1;
        while (dur_index < 0) dur_index += dur_vals.length;

        temp = (v - min_val_lin) / diff_val_lin;
        temp = (temp*1.8)-0.9;
        dur_vals[dur_index] = temp;
        temp_l2 = (Math.log(v)/log2);
        temp_l2 = (temp_l2 - min_val_log2) / diff_val_log2;
        temp_l2 = (temp_l2*1.8)-0.9;
        dur_log2[dur_index] = temp_l2;

}
set_next_dur.local = 1; //private.

function set_next_per(v)
```

```
{
	var temp, temp_l2;

	per_index -= 1;
	while (per_index < 0) per_index += per_vals.length;

	temp = (v - min_val_lin) / diff_val_lin;
	//post("period val: " + temp + "\n");
	temp = (temp*1.8)-0.9;
	per_vals[per_index] = temp;
	temp_l2 = (Math.log(v)/log2);
	temp_l2 = (temp_l2 - min_val_log2) / diff_val_log2;
	temp_l2 = (temp_l2*1.8)-0.9;
	per_log2[per_index] = temp_l2;

}
set_next_per.local = 1; //private.

function setscale(v)
{
	scale = v;
	//post(scale + " scale value\n");
	bang();
}

function in_describe(num)
{
	switch(num)
	{
		case 0:
			assist("duration values");
			break;
		case 1:
			assist("period values");
			break;
	}

}

function constrain(v,min,max)
{
	var nv = Math.min(Math.max(min,v),max);
```

```
	return nv;
}
constrain.local = 1; //private.

//
// User Interaction
//

function onresize(w,h)
{
	// erase background
	sketch.glclearcolor(vbrgb);
	sketch.glclear();
	draw();
	refresh();
}
onresize.local = 1; //private
```