Integrating analytical with compositional processes

Pandelis Diamantides
May 2007

Masters Thesis - Institute of Sonology
Royal Conservatory, The Hague

Abstract

This thesis is about integrating analytical with compositional processes.

In music cognition research several models have been proposed regarding the perception of musical structure. Those models informed the design and implementation of several analytical systems.

Frequently, research concerning the mechanisms of perceptual organization catches the attention of composers. Neither the idea of exploring those mechanisms, nor the urge to integrate analytical with compositional processes is something new.

The case of analyzing streams of midi numbers arrived from keyboard-based instruments resides in bygone eras. Onset detection algorithms that operate directly on audio signals are by now common practice. We present implementations, encapsulated as max/MSP abstractions, as part of our own attempt to integrate analytical with compositional processes.

However, the suggestion that analytical processes could be also suitable for composition, is under question.

Acknowledgements

I would like to thank:

Paul Berg for his guidance throughout this study. Peter Pabon for providing his insight into sound analysis and Renee Timmers for reading and commenting on parts of this thesis.

A special thanks goes to my parents Georgia and Diamantis Diamantides. Without their continuous and never ending support this study was never about to be finished.

An intimate thanks goes to Jessica Knijnenburg for her love and support throughout this year.

Contents

1. Introduction 8
2. Attempts to integrate analytical with compositional processes 11
   - 2.1. Temporal gestalt perception in music 12
   - 2.2. Interactive music systems 14
   - 2.3. Hybrid interactive music systems 16
   - 2.4. Experiments in musical intelligence 17
   - 2.5. Creating music by listening 18
3. Music cognition and computer-assisted composition 19
   - 3.1 Machine listening 20
     - 3.1.1 Onset detection 21
     - 3.1.2 Beat induction 23
   - 3.2 Machine learning 24
     - 3.2.1 Machine learning methodology 25
   - 3.3 Machine Composing 26
4. Segmentation schemes and musical structure 27
   - 4.1 Musical segmentation 28
     - 4.1.1 Gestalt grouping principles 29
     - 4.1.2 Grouping Preferences Rules 30
     - 4.1.3 A signal-based approach 32
   - 4.2 Musical audio signal segmentation 33
     - 4.2.1 Event segmentation 34
     - 4.2.2 Classification based on similarity 35
     - 4.2.3 Concatenative sound synthesis 36
5. Implementations 37
   - 5.1 max/MSP abstractions 38
     - 5.1.1 onset~ 39
     - 5.1.2 event~ 40
     - 5.1.3 groups 41
     - 5.1.4 concatenate~ 42

5.2 Example patches 43
5.2.1 EventCutUp 44
5.2.2 EventGroup 45
6. Discussion 46
6.1 Argumentum 46
6.2 Conclusions 49
6.3 Suggestions 50

Appendix: max/MSP abstractions and patches 51

References 62

List of figures

figure 1. onset~ 52

figure 2. onset2~ 53

figure 3. Offset~ 54

figure 4. event~ 55

figure 5. Location_Duration 56

figure 6. groups 57

figure 7. EventCutUp 58

figure 8. concatenate~ 59

figure 9. groups.help 60

figure 10. onset2~.help 61

Chapter 1:

Introduction

”Take care of the sense, and the sounds will take care of themselves”.
(Lewis Carroll, 1865-1871/1944, p.133)

Those charming words of Lewis Carroll appear in the opening of Dowling and Harwood’s (1986) book ’Music Cognition’. It is not a coincidence that I use the same words to start this thesis. However, my scope begs to differ:

listeners might benefit by taking care of the sense;
composers should take care of the sounds by themselves.

While analytical systems improve incrementally, is a legitimate time to ask: *what their function in the realm of composition might be*?

Frequently, research concerning the mechanisms of perceptual organization catches the attention of composers. Formalizations of these mechanisms into computer programs became essential components of music analysis systems.

Attempts to integrate analytical with compositional processes is not something new. There is a whole body of literature documenting the development of several different approaches.

The use of *Temporal Gestalts* (Tenney & Polansky 1980) in music analysis and composition is a paradigm of such an attempt that dates back to the early 60’s (Tenney 1961). Coupled with *Grouping Preferences Rules* (Lerdahl & Jackendoff 1983) are often incorporated into analytical systems.

However*, the -implicit or explicit- suggestion that ’analytical processes could be also suitable for composition, is arguable*.

Machine listening, learning and composing coordinate a broad area where analytical and compositional processes are called to operate altogether. Algorithms designed in a way that enables them to function in real-time, together with technological advances that offered computational power in personal computers, make possible implementation of models that provide the opportunity to carry analysis on stage. By attaining the computational speeds needed to execute compositional algorithms in real time, current computer systems are able to modify their behavior as a function of input from other performing musicians (Rowe 1993). Music cognition impacted on the development of interactive music systems, especially on their ability to function in ensembles including human performers. Aligned with music theory and artificial intelligence form the concept of machine musicianship (Rowe 2001). Those systems offered musicians the opportunity to become more engaged in compositional processes and bring to an advance level existing compositional processes, especially those that include improvisation (Rowe 1996, 1999).

Challenging analytical systems that operate on symbolic level, signal-based solutions model the life cycle of listening, composing and performing, turning the machine into an active musician, instead of simply an instrument (Jehan 2005). 'Event-synchronous' analysis (Jehan 2004) has been proposed as an antidote to brute force frame-based approaches in undertaken analytical tasks. Likewise, analysis based on a generic music cognition framework that combines both machine listening and machine learning, augments the standard pure signal processing approach of music analysis. It was hypothesized that creating music could share the same knowledge as acquired from a uniform analysis procedure based on perceptual listening and learning. Indeed, technologically advanced, however, as other example-based approaches (Cope 1991), are not compelling aesthetically.

Analytical systems with audio capture and reuse facilities extended the aesthetical outcome of interactive music systems, in an attempt to manifest hybrid systems i.e. interactive music systems that take as their input an audio signal alone, and yet also involve symbolic reasoning on extracted sound objects (Collins 2007).

Data-driven concatenative sound synthesis (Schwarz 2004) uses databases of source sounds to synthesize a sound or a musical phrase. This data-driven approach -as opposed to a rule-based approach- takes advantage of the information contained in the many sound recordings. Usual synthesis methods are based on a model of the sound signal. It is very difficult to build a model that would preserve all the fine details of sound. Concatenative synthesis achieves this by using actual recordings. Those methods are successful in synthesizing naturally sounding transitions.

However, the dream of developing tools, which enable the composers to concentrate, rather on high-level compositional questions than on low-level technical problems (Essl 2004) is mostly out of reach by current implementations of analytical tools. Although some problems have to do with the current state of the art, so it is possible that in the future they will be solved, misleading assumptions that by modeling the perception and cognition of humans one will be also able to use these implementations as novel compositional techniques should be avoided.

There is a whole body of literature documenting tools informed by the cognition of listening: onset detectors, beat trackers, event segmentation and grouping algorithms; are just some of the most common ones. Motivating are the ways that those tools combined towards automatic or semi-automatic systems. Especially the ones, that there design and implementation enables them to operate in real-time in response to musical input, within the context of interactive performances; as well as in the studio, where by transforming common musical computer applications into intelligent interlocutors provided the opportunity to speed up compositional processes.

Implementing models that attempt to resemble human perception and cognition is far for being trivial. Beat induction is an example of a simple cognitive task that is nevertheless hard to model computationally (Desain & Honing 1994). As result it has been an important issue in interactive computer music, with many different approaches explored. The systems that were developed all use a more or less pragmatic approach. Indeed, concerning music, most of the times a pragmatic approach offered simple solutions to complicated problems by using humans to guide systems throughout analytical processes, e.g. a human 'taps tempo' for the system as a reference quantization grid.

Analytical systems, under certain limitations concerning musical context, behave reasonably. Nevertheless, is difficult to deal with their deficiencies. Compact, while the same time descriptive, abstractions to deal with the output of simple analytical tools is often more efficient than sophisticated algorithms that their output is frequently 'data explosions'.

We will come back to the aforesaid issues and try to give a better-informed answer towards the integration of analytical with compositional processes in our discussion (chapter 6). In the following chapters (2, 3, 4 and 5) we will try to gain as much information as possible concerning our topic.

Chapter 2:

Attempts to integrate analytical with compositional processes

”The confusion between music theory and composition becomes even greater when analytic systems are described in detail followed by the suggestion that by making ad hoc changes to the system, it would also be suitable for composition”
(Paul Berg, 1996, p. 25)

Neither the idea of exploring the mechanisms that concern perceptual organization, i.e. the nature of relations among parts and wholes and how they are determined, nor the urge to integrate analytical with compositional processes is something new.

There is a whole body of literature documenting attempts to integrate analytical and compositional processes. Tenney (1961, 1980) suggested the use of Temporal Gestalts in music composition. Cope (1992) carried out his Experiments in Musical Intelligence. Rowe (1993) contributed a broad review of Interactive Music Systems and presented his own program Cypher. Recently, Collins (2007) picked-up the same idea and tried to expand it to the signal domain, constructing hybrid interactive music systems. Jehan (2005) developed a framework for Creating Music by Listening.

Let us briefly describe those attempts and consider their contributions in music composition.

2.1 Temporal gestalt perception in music

In the early 1930s Gestalt Perception theorists (Wertheimer 1923) was rejecting the structuralists' proposal that complex perceptions were constructed from atoms of elementary sensations and unified by associations due to spatial and temporal contiguity. Those assumptions were rejected, arguing that perception is holistic and organized due to interactions between stimulus structure and underlying brain processes (Palmer, 1999).

From 1960s through 1980s composer/music theorist James Tenney and composer/programmer Larry Polansky were working upon the use of temporal gestalts on the way to achieve 'perceptually effective formal structures' concerning music composition (Tenney & Polansky, 1980).

Assuming that the factors involved in temporal gestalt perception are objective in some extend, while bearing some measurable relation to the acoustical properties of the sounds themselves: is the effect of those factors strong enough that one might be able to predict where temporal gestalts' boundaries will be perceived (of course, if one knows the nature of the sound events that will occur)? In an effort to provide some tentative answers to such questions they designed and implemented in a computer program a model based on a hypothesis of temporal gestalt perception in music stated as follows:

'A new temporal gestalt at the next higher level will be initiated in perception whenever a temporal gestalt occurs whose disjunction (with respect to the previous temporal gestalt at the same hierarchical level) is greater than those immediately preceding and following it' (Tenney & Polansky, 1980, p. 217).

In spite of the limitations of the model, the hypothesis of temporal gestalt perception on which is based on was suggested as a plausible formulation of an important principle of musical perception; due to the degree to which its results correspond to temporal gestalt boundaries arrived at by other means (i.e. group boundaries suggested by a human observer).

What interest us, relating to the topic of this thesis, is that they asserted that their model might have useful applications for the composer, since it can be used to create '*perceptually effective formal structures without recourse to traditional devices - tonal or otherwise*' (Tenney & Polansky, 1980, p. 235).

Next to the observation that other compositional methods, (serial, aleatoric and stochastic) frequently result in textures, which are statistically homogeneous at some fairly low hierarchical level (where a typical negative response to this kind of formal situation is that, although everything is changing, everything remains the same) is the claim that their model suggests a technique for controlling this aspect of musical form.

They persist by giving a technical remedy to achieve 'ergodicity' as described elsewhere (Tenney 1961) and they related this to temporal gestalts; by demonstrating that a piece becomes 'ergodic' as soon as a hierarchical level is reached at which the states of successive temporal gestalts are indistinguishable (i.e. at that level at which the mean intervals between successive temporal gestalts are effectively zero).

In general, according to Tenney and Polansky, this can be shown to depend on the degree to which parametric ranges are constrained at the lower levels, that is, the more the total available range in some parameter is 'used up' at that given level, the smaller will the average effective differences be between temporal gestalts at that level. Thus, the more quickly will the texture approach 'ergodicity' at the next higher level. The technical remedy for this is to distribute the total available ranges more evenly over as many hierarchical levels as needed to achieve the formal structure intended.

Tenney and Polansky's model ignores musical forms that rely on vertical density, timbre or are multi-layered. As will be discussed this is really taking simplification too far.

On the other hand, models following the path of temporal gestalt perception, proven to be useful in real-time musical circumstances, typically during the course of a live performance, where automatic detection of phrase boundaries is usually needed. In general, asserting group boundaries in a lower-level, is a first-step for analysis algorithms that enable a computer-based performance system to be aware of the musical context where is called to contribute.

2.2 Interactive music systems

Interactive music systems, i.e. those able to provide plausible response to a musical input, opened up a domain where analytical and compositional processes are combined in an intriguing way.

Rowe (1993) reviews a wide range of interactive systems including his own, Cypher. This system consists of two major real-time components, the 'listener' (analysis) and the 'player' (composition). The listener component analyzes incoming musical data (MIDI) and the player component responds to this information by generating new relevant musical material. Each component consists of interrelated agents, which constitute agencies, in the Minskian sense (Minsky 1988).

Analytical processes in Cypher are called to operate on hierarchical levels: first, classify incoming musical events across a number of features (e.g. loudness, speed, and register); and second, characterize the behavior of these features within the current phrase, as being regular or irregular.

The composer constructs rules using the output of this analysis to determine which compositional algorithms will be invoked in response to which analyses and how the behavior of the compositional algorithms will change over time. One should note that the listener and player modules are relatively independent.

As Cypher attempts to tackle many aspects and levels of musical analysis in real-time, it is led to only generating a simplified analysis of the input musical structures (especially as far as higher-level organization is concerned). In the trade-off between interactive real-time pragmatic efficiency and elaborate exhaustive analytic expressiveness, this system is biased towards the former (Cambouropoulos 1998).

Interactive music systems meant to perform onstage either in conjunction with human players or completely autonomously. Thus, a novel and engaging form of interaction between humans and computers introducing real-time algorithmic composition in works including improvisation was created (Rowe 1996).

Because interactive music systems derive control parameters from an analysis of live performance, they can generate material based on analyses of improvisation as easily as they can on analyses of notated music. Such systems become a ligature connecting improvisation to notated composition, as the same processes used to govern the notated music can be employed to generate new improvisations in real time.

This possibility is an expansion of the domain of composition.
By delegating some of the creative responsibility to the performers and a computer program, the composer pushes composition up (to a meta-level captured in the processes executed by the computer) and out (to the human performers improvising within the logic of the work).

An interesting effect of this delegation is that it requires a very detailed specification of the musical decisions needed to produce a computer program at the same time that the composer cedes a large measure of control over musical decision-making to the human improviser. The resulting music represents a new kind of composition at the same time that it necessitates new kinds of performance skills.

Nevertheless, computer systems cannot function in improvisation unless they are programmed with ways to make some sense of the context around them and to react in accordance with that sense. This brings one to the topic that we refer to as *machine musicianship,* i.e. the realization of concepts concerning human musicianship on a computer program, coupled with emulations of human cognitive processes (Rowe 2001). Some of those processes along with their computer-based simulations I will describe in the next chapter[1].

There are compelling musical reasons to emulate human musicianship with computers. Machine musicianship can strengthen and extend human ways of making music. Making computer programs able to recognize and reason about human musical concepts enables the creation of applications for performance, education, and production that resonate with and reinforce the basic nature of musicianship. Implementing musicianship on a computer program allow us to augment human musicianship with processes and representations that only a computer could provide (assuming that those implementations resemble in some extend human cognitive processes and musicianship). For example, a complete record of the program's 'listening experience' is immediately available and can be used both to evaluate the computer's performance and also to direct further analysis.

Using processes in performance that change their behavior according to an analysis of other player's music, however, was never possible before the advent of interactive music systems. Such systems therefore engender a realm of composition that was unknown only a few decades ago. On the other hand, the musical values evinced in interactive compositions are ultimately the same as those underlying a string quartet. By transferring musical knowledge to a computer program and compositional responsibility to performers onstage the composer of interactive works explores the creative potentials of the new technology at the same time that (s)he establishes an engaging and fruitful context for the collaboration of humans and computers (Rowe 1999).

[1] see chapter 3

2.3 Hybrid interactive music systems

Collins (2007) made an effort to build hybrid interactive music systems, i.e. systems that take as input an audio signal alone, and yet also involve symbolic reasoning on extracted sound objects; where those sound objects, being the content of a database are used by compositional algorithms to form a plausible response.

Onset detection[2] algorithms are utilized for the real-time extraction of sound objects with symbolic attributes from an audio signal. Event segmentation[3] has been exploited in order to form the databases of events and allow symbolic reasoning over the extracted sound objects. In particular, a real-time multi-feature onset detector was implemented, for baroque recorder, which combines pitch, and amplitude cues. However, the main focus has been percussive instruments, which are a more manageable case because of the discrete nature of their sound.

The aforesaid interactive systems developed for a range of musical styles and instruments, all of which attempt to participate in a concert by means of audio signal analysis alone. Machine listening, and the hypothetical modeling of central auditory and cognitive processes, is utilized in those systems to track musical activity. Whereas much of this modeling is inspired by a bid to emulate human abilities, strategies diverging from plausible human physiological mechanisms are often employed, leading to machine capabilities, which exceed or differ from the human counterparts.

All the technology produced is intended for realtime concert use. In order to exploit processes that underlie common musical practice, beat tracking is investigated, allowing the inference of metrical structure that can act as a coordinative framework for interaction.

Psychological experiments into human judgment of perceptual attack time and beat tracking to ecologically valid stimuli clarify the parameters and constructs that should most appropriately be instantiated in the computational systems.

The production of hybrid systems is a result of having both low-level feature and higher-level sound object extraction and symbolic manipulation in one system. Musical agents that can interact with human musicians in concert situations were reality already before this research.

Still, the extent to which they themselves embody human-like capabilities can be called into question. '*They are perhaps most correctly viewed, given their level of artificial intelligence technology, as 'projected intelligences', a composer's anticipation of the dynamics of a concert setting made manifest in programming code'* (Collins 2007, p.6).

[2] see section 3.1.1
[3] see section 4.2.1

2.4 Experiments in musical intelligence

Experiments in Musical Intelligence (EMI), an example-based system developed by Cope (1991, 1992), is a computer model of musical composition based on style analysis of a composer's body of works. This system focuses on the replication of works in the style of an individual composer, which is grounded on the observation that composers tend to reuse musical patterns throughout their corpus of compositions.

The system requires at least two compositions in a similar style from which it induces 'musical signatures' and rules for composition, mainly statistical analysis (Cambouropoulos 1998). The input works are presented as separate lists of phrases of MIDI note numbers, i.e. a preliminary grouping of works is externally defined at the level of phrase structure. Analysis is a series of mathematical subprograms that compute percentages of certain aspects of music such as voice leading directions, use of repeated notes, triad outlining, leaps followed by steps, etc. (Cope 1993). A signature, being the output of such an analysis, is a set of contiguous intervals found in more than one work by the same composer (Cope 1991).

In the composition phase, 'the program 'fixes' signatures to their same locations in an otherwise empty form based on the form of the first of the input works'. The intervening spaces between signatures are composed based on the rules discovered by the statistical analysis.

EMI relies on a grammar, which follows an idiom-specific protocol of musical functions and hierarchic relations, primarily a classical tonal protocol. Although one may vary the interpreter protocols they are defined based on previously acquired musical knowledge and have an overall 'tonal' outlook (Cambouropoulos 1998). The works generated by this model resemble quite successfully music in the style, for instance, of Bach, Mozart, Brahms, e.t.c.

Although, this example-based approach may increase our knowledge of certain styles or make other music theory contributions, the results concerning music composition are not particularly compelling aesthetically (Berg 1996).

2.5 Creating music by listening

Tristan Jehan developed a system, subsequent to the observation that '*machines have the power and potential to make expressive music on their own' (*Jehan 2005, p.3). In his study models the process of creating music using experience from listening to examples. In an attempt to turn the machine into an active musician, models the life cycle of listening, composing, and performing.

An interesting music cognition framework has been introduced, that results from the interaction of psychoacoustically grounded causal listening, a time-lag embedded feature representation and perceptual similarity clustering. A bottom-up signal-based analysis, which combines perceptual and structural modeling of the musical surface, intends to be generic and uniform by recursively revealing metrical hierarchies and structures of pitch, rhythm, and timbre.

Jehan's system constitutes a practical implementation of machine intelligence for music analysis and synthesis where the data representation resulting from the analysis can be common to the synthesis data: *'It was indeed hypothesized that synthesis could share the same knowledge as acquired from a uniform analysis procedure based on perceptual listening and learning. Our analysis is based on a generic four-stage music cognition framework that combines both machine listening and machine learning, augmenting the standard pure signal processing approach of music analysis.'* (Jehan 2005, p.114).

Outstanding is that this work constitutes a generic perceptual model of music cognition, rather than scattered and self-contained algorithms and techniques. The system enables a range of original manipulations including song alignment, music restoration, cross-synthesis or song morphing, and ultimately the synthesis of original pieces.

Rather as a contribution to machine music intelligence than a contribution on the aesthetic and artistic fronts; has been shown that a computer program can close the cycle of listening, composing, and performing music through audio signals.

Chapter 3:

Music cognition and computer-assisted composition

”The use of brain-style computational systems, then, offers not only a hope that we can characterize how brains actually carry out certain information-processing tasks but also solutions to computational problems that seem difficult to solve in more traditional computational frameworks.”
(David Rumelhart, 1989 p.209)

In this chapter we consider tools informed by the cognition of listening; those components that constitute systems capable of taking in information about their musical environment, forming internal representations of it, and manipulating these representations to select and execute actions.

In music cognition research several models have been proposed that are concerned with the perception and cognition of musical structure. Those models informed the design and implementation of several analytical tools that assist composition and performance in various levels of progress.

Given the limited space of a thesis, we restrict ourselves to common tools relevant to the realm of ‘intelligent signal processing’. Those are *onset detection* and *beat tracking*; and in a somewhat higher-level: *machine listening*, *learning* and *composing*.

Event segmentation and *grouping* are more related to the implementations we present in chapter 5. Therefore will be described in more detail in a dedicated chapter[4]

[4] chapter 4

3.1 Machine listening

The most widespread analytical tools are components of what we refer to as 'machine listening', i.e. '*the simulation of human peripheral auditory abilities, and the (hypothetical) modeling of central auditory and cognitive processes, which is utilized in analytical systems to track musical activity*' (Collins 2007, p.6).

Research in machine listening is concerned with the implementation of algorithms capable to imitate the way humans perceive music. There are three major machine listening approaches: the physiological approach, which attempts to model the neurophysical mechanisms of the hearing system; the psychoacoustic approach, rather interested in modeling the effect of the physiology on perception; and the statistical approach, which models mathematically the reaction of a sound input to specific outputs.

Auditory models, i.e. simulations of human hearing, are often implemented as the first level of a machine listener. An interesting approach to machine listening is the auditory spectrogram used in Jehan's (2005) model: *'the goal of the auditory spectrogram is to convert the time-domain waveform into a reduced, yet perceptually meaningful, time-frequency representation. We seek to remove the information that is the least critical to our hearing sensation while retaining the most important parts, therefore reducing signal complexity without perceptual loss' (*Jehan 2005, p.43). Even so, listening is a much more complex process than hearing. Therefore, approaches to build listening capabilities to machines imply classification algorithms[5].

One of the main reasons for implementing listening capabilities into a system is to make it able to understand -in some extent- its musical environment. By understanding a given musical environment we mean not only extracting the characteristics of sounds (feature extraction), but also the various ways that those are combined as to form musical structures.

Then again, there is not one way that humans listen to music; and there is not one way to build listening capabilities into a machine. Assessments regarding the question of what a machine can hear, rely upon the musical context in which a system is expected to function. It goes without saying that a machine listener also depends on the current technological achievements, e.g. computation speed.

Machine listeners are crucial components of interactive (responsive) music systems since the responsiveness of those systems requires them to make some interpretation of their input (Rowe 1993).

[5] see section 4.2.2

3.1.1 Onset detection

We call onset detection: the detection of the beginnings of discrete events in audio signals. Onset detection plays an important role in the computational segmentation and analysis of musical signals and is a fundamental requirement of machine listening work.

A lot of research in onset detection has been carried out in recent years and several approaches have been proposed[6]. We highlight Klapuri's (1999) onset detection system, which builds upon the use of relative difference function and application of the psychoacoustic models of intensity coding. Experimental results show that the aforementioned system exhibits a significant generality in regard to the sounds and signal types involved. This was achieved without higher-level logic or a grouping of the onsets. Nevertheless, as the author of the system indicates, in the case of musical signals, an additional higher-level analysis would still significantly improve the accuracy of the system.

There is a practical benefit in talking of discrete objects, given that the aim is to interpret or comprehend the sound. The alternatives would be either to process the sound stream all at once, or else in fixed-length sections. Most sound streams are too extended to permit the former approach, since one needs some results of the interpretation before the sound stream finishes. The latter approach would require the sections to be shorter than the sorter element of the sound, and would present problems when a section contains parts of more than one sound element (Smith 1994). We will come back to the discussion about using discrete sound objects later[7].

Most onset detectors work in a way that loosely follows the early stages of a human hearing model. The incoming audio signal is split into a set of fixed filters over the most sensitive parts of the human hearing range, and for each a form of temporal integration of energy is applied. In an alternative approach, closely related to 'classic' digital signal processing, a frequency domain transform is applied using Fast Fourier Transform (FFT), and features sought over frames from an examination of changing phase vocoder information, i.e. phase and amplitude of FFT bins. Derivatives of these signals may be taken rather than the pure values. A second stage copes with the selection of peaks in the smoothed envelopes for signal energy in each band, by some absolute or adaptive threshold, and by considering the combination of results across subbands (Collins 2005b).

[6] for a comprehensive comparison of onset detection algorithms see Collins (2005a)

[7] see section 4.2.1

Cognitive models for onset detection may aid musical event segmentation in the manner of a human observer. Nevertheless, they may not give the best solution to match the discovery of transient sound events. There are differences between detecting onsets, as a human observer would function in realtime and applying onset detection to segment events for digital editing purposes. The former may mimic the cognitive procedures of human listeners, and has dividends in machine listening while the latter requires segmenting events as fast and as accurately as possible. The nature of the sound events to be detected and the intended application determines the appropriate detection strategy (Collins 2007).

The case of analyzing streams of midi numbers arrived from keyboard-based instruments resides in bygone eras. Onset detection algorithms that operate directly on audio signals are by now common practice. However, onset detectors not always perform as expected. Particularly, problems arise while one attempts to detect the sound onsets one-by-one. On the other hand, the performance of onset detectors when operate in parallel with higher-level analytical processes, e.g. beat tracking or event grouping, is reasonable.

3.1.2 Beat induction

Beat is a perceptually induced periodic pulse that defines tempo. The ability to infer beat and meter from music is one of the basic activities of musical cognition. After having heard only a short fragment of music we are able to develop a sense of beat and meter and tap our foot along with it. Even if the music is rhythmically complex, containing a range of different time intervals and probably syncopation, we are capable of inferring the different periodicities of it and synchronizing to them.

However, beat induction is an example of a simple cognitive task that is hard to model computationally. Therefore, in music cognition research several theories have been proposed that are concerned with the perception of different types of temporal musical structure (Desain & Honing 1994).

Models of beat induction presented to date have been based on various computational formalisms. These include symbolic, statistical approaches, optimization approaches, control theory, connectionist models, and oscillator models. All these models rely solely on the temporal structure. Scheirer (1998) presented a model that is taking into account features related to pitch. The aforementioned model uses audio input that is passed though a bank of band-pass filters. Noteworthy is also the work of Toiviainen and Snyder (2000) that explores the time-course of pulse sensation and its dependence on various musical features, such as onset time structure, pitch height, and harmonic structure.

Desain and Honing (1994) assess that both cognitive and technological approaches have not been able to arrive at a general, robust beat extraction method: '*Many models have evolved and several complicated computer music systems exist that behave reasonably, but none of the systems has attained the generality and robustness strived for'*.

Subsequently, and in agreement with the observation that downbeat cannot be computed only from signal processing, and requires training, Jehan (2005) presented an implementation of a general downbeat predictor.

3.2 Machine learning

Learning makes possible modification of behavior in response to the environment.

The goal of machine learning is to build computer systems that can adapt and learn from their experience. Learning takes place as a result of the interaction between the program and the world, and from observation by the program of its own decision-making processes.

There are a wide variety of algorithms and techniques but there is no ideal one. Different learning techniques have been developed for different performance tasks. Results usually depend on the problem that is given, the complexity of implementation, and time of execution.

Machine learning algorithms search a space of candidate classifiers for one that performs well on the training examples and is expected to generalize well to new cases. Learning methods for classification problems include decision trees, neural networks, rule-learning algorithms, nearest neighbor methods, and certain kinds of Bayesian networks.

There are two fundamentally different theories of machine learning. The classical theory takes the view that, before analyzing the training examples, the learning algorithm makes a 'guess' about an appropriate space of classifiers to consider. The algorithm then searches the chosen space of classifiers hoping to find a good fit to the data.

The Bayesian theory takes the view that the designer of a learning algorithm encodes all of his or her prior knowledge in the form of a prior probability distribution over the space of candidate classifiers. The learning algorithm then analyzes the training examples and computes the posterior probability distribution over the space of classifiers. In this view, the training data serve to reduce our remaining uncertainty about the unknown classifier (Dietterich 1999).

3.2.1 Machine learning methodology

There have been described three primary forms of machine learning. Those are: *supervised learning, unsupervised learning* and *reinforcement learning*. These methods assume that a set of examples or instances is known (Kasabov 1996).

In supervised learning the training examples comprise 'input' and the desired 'output'. Training is performed until each 'input' example is associated to its corresponding and desired 'output'. A large amount of training examples is required.

In unsupervised learning, or clustering, only examples comprise 'input' are supplied. The system learns some internal features of the whole set of all the 'input' presented to it and forms corresponding clusters. The number of clusters can be specified in advance.

Reinforcement learning is based on presenting 'input' and looking at the 'output' that is produced by the system. If the 'output' is considered 'right', then the association will be reinforced.

We can review the abovementioned methods using a musical example: let's suppose that a machine is expected to learn how to recognize the timbre of some musical instruments. Following the unsupervised learning method we were about to present to the analytical system in use, a large database of musical instrument recordings where we know their origin. In an unsupervised learning approach, several clusters would be formed and we would expect, that each one would represent different timbre classes. By the use of a reinforcement learning method, we were about to 'reward'[8] the system every time was about to recognize correctly an instrument or at least was about to approach a correct inference, and 'punish' it every time that was about to be unsuccessful.

[8] 'Reward' and 'punishment' in machine learning is a whole issue by itself and is nothing to concern us about it here.

3.3 Machine Composing

There are hardly any algorithms informed by the cognition of listening that are devoted to music composition. Part of the reason for that may stem from the fact that composers are engaging in quite different activities when they work. Another reason is that it is much harder to test what is going on cognitively during composition (Rowe 2001).

The available literature describes for the most part *transformative* algorithms, i.e. algorithms that operate upon analyzed input material and output various musically plausible transformations.

Another common practice is to map the output of analytical processes to the input of any kind of *generative* algorithms. This way of mapping has been characterized as 'arbitrary' or 'intuitive', an issue that *raises questions about the necessity of analysis, as soon as the same results could be achieved by 'typing in' the analysis data*.

However, a musical situation that this might be inconvenient or even impossible to do so ('type in' analysis data) is an interactive live music performance. For this reason we will use that kind of a musical situation, that involves both analytical and compositional processes, to illustrate the function of the two aforesaid classes of algorithms: lets suppose that an analytical system 'listens' intensively to a musical instrument and continuously produces data. Those data include locations and durations of salient events in the captured audio signal. In a somewhat higher-level of analysis those events are grouped as to form musical phrases. The compositional processes of our system are called to produce an output every time silence of more than 750ms is detected;

using some transformative algorithm,

retrieve a phrase of minimum 7 events,
swap event durations,
transpose event pitch randomly one semitone up or down,
reverse event order
playback

using some generative algorithm,

start generating random values every 100-230 ms
constrain random values between the lowest and the highest value
(lowest and highest pitch values from real-time analysis of input)
generate random midi pitch numbers within those changing boundaries
send values to midi-to-frequency converter

Chapter 4:

Segmentation schemes and musical structure

"If all first-order elements were indiscriminately linked together, auditory shape recognition could not be performed. There must, therefore, be a set of mechanisms that enable us to form linkages between some elements and that inhibit us from forming linkages between others" (Diana Deutsch 1999, p.299)

During acoustical communication the acoustical signal is an intermediate phase, presenting two points of contact: we can derive from it properties of the sound source; and we can attempt to predict what the effect of the signal on the receiver will be. In the last case the analysis becomes a simulation of the demodulation process that takes place in the perception of acoustical signals (Tempelaars 1996).

In listening to a piece of music, the human perceptual system segments the sequence of notes into groups or phrases that form a grouping structure for the whole piece. Two widely accepted grouping principles in music are the Gestalt principles of proximity and similarity and the higher-level principle of melodic parallelism. Nevertheless, it was shown by Deliège (1987) that listeners tend to prefer grouping rules based on timbre over other rules, i.e., melodic and temporal and by Lerdahl (1987) that musical structures could also be built up from timbre hierarchies.

While most models also incorporate higher-level grouping phenomena, such as melodic parallelism and harmony, these phenomena remain often unformalized. As a result, these models have not been evaluated against large sets of musical data. Only a few, hand-selected passages are typically used to evaluate these models, which questions the objectivity of the results (Bod 2000).

Musical audio signal segmentation avoids having to distinguish between 'score-based' properties of a musical event and 'expressive' properties in considering perception. Event segmentation is used as a building block for automated music analysis and classification algorithms using as input segmented objects (Tzanetakis & Cook 1999). Segmentation results combined with the calculated feature vectors could be used as an intermediate representation for further higher-level analysis (Scheirer 1998; Jehan 2005). Other applications include automatic transcription and annotation.

Moreover, there is an increasing interest in incorporating automated event capture and reuse algorithms in interactive music systems (Brossier, Bello & Plumbley 2004; Aucoutourier & Pachet 2006; Collins 2007).

4.1 Musical segmentation

Musical segmentation is the process by which musical events are organized into groups, i.e. *event grouping*. This should not be confused with the detection of those musical events in a musical audio signal, i.e. *event segmentation*[9].

The *Gestalt grouping principles* and the *Grouping preferences rules* became essential in almost any attempt to induct musical phrases while analyzing a musical stream. Often are incorporated into computer-based musical systems that aim to automate analysis prior to compositional processes.

[9] see section 4.2.1

4.1.1 Gestalt grouping principles

Most models of event grouping use the Gestalt principles of proximity and similarity (Wertheimer 1923) to predict the low-level grouping structure of a piece: grouping boundaries preferably fall on larger inter-onset-intervals, larger pitch intervals, etc. (Tenney& Polansky 1980; Lerdahl & Jackendoff 1983; Cambouropoulos 1998). Those principles have been considered as formalizations of intervallic distances, parallelism, meter, harmony or other musical phenomena.

Gestalt principles state that objects closer together (proximity) or more similar to each other (similarity) tend to be perceived as groups. These principles have been used as a basis for some contemporary theories of musical rhythm. Tenney (1961) discusses the use of the principles of proximity and similarity as a means of providing cohesion and segregation in 20th century music and, later, Tenney and Polansky (1980) developed a computational system that discovers grouping boundaries in a melodic surface. Musical psychologists (Bregman, 1990; Deutsch, 1999) have experimented and suggested how the Gestalt rules may be applied to auditory and musical perception and Deutsch and Feroe (1981) further incorporate such rules in a formal model for representing tonal pitch sequences. The grouping component of Generative Theory of Tonal Music (Lerdahl & Jackendoff 1983) is based on the Gestalt theory and an explicit set of rules is thereby described especially for the low-level grouping boundaries. The formulation of these rules has been supported by the experimental work of Deliège (1987).

In Tenney and Polansky's (1980) work, the principles of proximity and similarity are used as the basis for rules that govern grouping of elements, clangs, and sequences: '*an element may be defined more precisely as a Temporal Gestalt, which is not temporally divisible, in perception, into smaller temporal gestalts. A clang is a temporal gestalt at the next higher level, consisting of a succession of two or more elements, and a succession of two or more clangs-heard as a temporal gestalt at the next higher level constitutes a sequence'. Segment and Units term grouping at the next two higher levels.* Their system implements the idea that principles of proximity and similarity are two primary factors contributing to group formation in music perception. The rule related to proximity is defined as follows: *'in a monophonic succession of elements, a clang will tend to will be initiated in perception by any element which begins after a time-interval (from the beginning of the previous element, i.e., after a delay-line) which is greater than those immediately preceding and following it, other factors being equal'*. Their similarity rule is a generalization of their proximity rule: *'In a monophonic succession of elements, a clang will tend to initiated in perception by any element which differs from the previous element by an interval (in some parameter) which is greater than those (inter-element intervals) immediately preceding and following it, other factors being equal'*. Their approach also looking for intensifying discontinuities, i.e. differences between neighbors in which the middle element changes more than the others (Rowe, 2001).

4.1.2 Grouping Preferences Rules

Generative Theory of Tonal Music (GTTM) (Lerdahl and Jackendoff 1983) accounts for the intuitions of experienced listeners in the tonal idiom. The main components of their theory are grouping structure, metrical structure, time-span reduction and prolongation reduction. Grouping structure expresses a hierarchical segmentation of a piece into motives, phrases, and sections. Metrical structure expresses the intuition that the events of a piece are related to regular alternation of strong and weak beats at a number of hierarchical levels. Time-span reduction assigns to the pitches of the piece a hierarchy of 'structural importance' with respect to their position in grouping and metrical structure. Prolongation reduction assigns to pitches a hierarchy that expresses harmonic and melodic tension and relaxation, continuity and progression. The theory is developed in a rather formal manner and rules are divided into two distinct types: well-formedness rules that define possible structures and preference rules that specify descriptions that correspond more closely to listeners' intuitions. However, GTTM is by far the most systematic work used for the deduction of musical structure. Therefore, has been the object of empirical investigation (Deliège, 1987).

Lerdahl and Jackendoff proposed that grouping is essentially hierarchical property of music, and in their Grouping Well-Formedness Rules, they outline the formal conditions for hierarchical structure. Coupled with this, the Grouping Preferences Rules describe the conditions that determine which of a very large number of possible hierarchical segmentations of any passage of music are actually likely to be perceived by listeners. The preference rules do not rigidly determine the segmentation of any particular passage, but specify the various forces acting in any musical context, which may reinforce one another or compete, resulting in different segmentations for different listeners (Clarke, 1999). The Grouping Preferences Rules themselves consist of three components: formalized gestalt principles (principles of proximity in time, or change in pitch, duration, loudness, or articulation); more abstract formal concerns (principles of symmetry and the equivalence of variants of the same segment or passage); and principles related to pitch stability (Cambouropoulos 1998).

As we see in the following sample list of Grouping Well-Formedness and Preferences Rules, some rules are readily usable as they stand, while others present more difficulty, both in terms of formulation as well as application (Rowe 2001);

Grouping Well-Formedness Rules (GWFR):

GWFR1) Any contiguous sequence of pitch events, drum beats, or the like can constitute a group.
CWFR2) A piece constitutes a group.
CWFR3) A group may contain smaller groups.
CWFR4) If a group G1 contains part of a group G2, it must contain all of G2.
CWFR5) If a group G1 contains a group G2, then G1 must be exhaustively partitioned into smaller groups.

Grouping Preferences Rules (GPR):

GPR1) Strongly avoid groups containing a single event
GPR2) (proximity) Consider a sequence of four notes n1n2n3n4. All else being equal, the transition n2-n3 may be heard as a group boundary if
a) slur/rest: the interval of time from the end of n2 to the beginning of n3 is greater than that from the end of n1 to the beginning of n2 and that from the end of n3 to the beginning of n4, or if
b) attack-point: the interval of time between the attack points of n2 and n3 is greater than that between the attack points of n1 and n2 and that between the attack points n3 and n4.

Even though GTTM is not algorithmic, since competing structures could be generated from rules that are both legal and incompatible with one another, there have been several efforts to implement parts of the suggested rules algorithmically (Stammen and Pennycook, 1993).

The GTTM attempts to describe musical structure by adopting a stance that is influenced by linguistic theory. In doing so, it may be argued that it sometimes gives rise to formalisms that do not seem to reflect musical structure in the most adequate way. For instance, the well-formedness rules are unnecessarily rigid. It has been argued (Cambouropoulos 1998) that: *'strict well-formed tree-like structures should not be considered as the norm (with possible course of which appropriate features and segmentations are discovered, they do not offer a tractable algorithm for implementing this (except only in the simplest cases of surfaces consisting mostly of exact repetitions where the search space is sufficiently small)'*.

Ultimately, the GTTM is a theory of tonal music. Even so, there are aspects of the theory that are style-independent, especially the Gestalt-based grouping rules. Lerdahl (1989) attempts to adapt the GTTM for experienced listeners in atonal music, but presents experimental evidence that doesn't seem to support his proposal.

4.1.3 A signal-based approach

Tenney and Polansky put forward that: '*would be possible to extend the model*[10] *'downward' to sub-element levels, which would not only eliminate the tedious process of transcription now required to specify input data to the program, but also be far more accurate than this process can ever be, in representing the sounds as we actually hear them. Such an extension would involve analog-to-digital conversion of the acoustical signal into numerical 'samples' suitable for input to the computer program* (Tenney & Polansky 1980, p.35).

Todd (1994) developed a model of grouping that converges towards solutions that are often similar to those offered by Gestalt principles and GTTM, but is based rather on more explicit perceptual processes and has close parallels to documented properties of the auditory system.

The central principle of Todd's approach is the idea that the functioning of the auditory system can be seen as the operation of a number of energy-integrating low-pass filters with relatively small time constants, integrating acoustical energy over durations of the order of milliseconds or a few tens of milliseconds. At a somewhat higher-level, small groups are detected as relatively discrete packets of integrated energy over periods of around a second. Larger, and hierarchical superordinate, groups are detected by virtue of integrators using exactly the same process, but with correspondingly longer time constants. Looking for zero-crossings in the second derivative of the filter output can identify peaks in the output of these low-pass filters, and these peaks are plotted across all the filters in a multiscale assembly, a representation of events at a number of levels, and the grouping relationships between events is obtained.

An attractive feature of this model is that, because it is based on energy integration, it is sensitive to any changes in the acoustical signal that have consequences for the integrated energy level. This includes note duration, pitch, intensity, and even timbre and vibrato, so that the written value of any note and any expressive treatment that it receives in performance all contribute in an undifferentiated manner to the integrated energy level that is output of the filter.

The virtue of this is that it avoids having to distinguish between 'score-based' properties of a musical event and 'expressive' properties in considering perception- a distinction that is anyway meaningless for all those musical cultures that do not use notational systems - which is the majority of world music (Clarke 1999).

[10] Tenney and Polansky's model (see section 2.1)

4.2 Musical audio signal segmentation

The alert reader will no doubt suspect that the segmentation technique we are going to draw attention to is the onset- based segmentation, i.e. event segmentation; primarily because techniques such as onset detection are able to provide a much more informative musical audio signal segmentation than any fixed length segmentation due to the fact that sounds do not occur in fixed length segments (West & Cox 2005).

4.2.1 Event segmentation

In their article 'Musical content analysis through models of audition' Martin, Scheirer and Vercoe (1998) argued that music analysis systems should be built for and tested on real music and be based on perceptual properties rather than music theory and note-level transcriptions. This aligns with the essential idea behind event segmentation, that is to divide the audio signal into semantically meaningful units, in a similar way to the decomposition produced by human perception of audio, i.e. into individual sounds (West & Cox 2005). Following a bottom-up methodology peaks in the positive differences in the audio signal envelope correspond to onsets in the audio stream, i.e. the beginning of musical events.

From an ecological viewpoint, we try to associate events with sounds in order to understand our environment (Bregman 1990). The characteristics of sound sources tend to vary smoothly in time. Therefore abrupt changes usually indicate a new sound event. Significant changes of multiple features usually indicate event boundaries. It has been suggested (Tzanetakis & Cook 1999) that those features contain enough information to be useful for automated event segmentation.

An event is considered meaningful if it does not contain any noticeable abrupt and is defined by its onset and offset boundaries.
Onsets can be found by extracting the local maxima in the resulting function and by searching the closest local minima in the loudness curve (Smith 1994). A zero cross correction might be applied to avoid clicks in reuse of the events.

While many music-oriented applications require real-time functionality, little has yet been done to address the issue of real-time extraction of events. Segmenting sound events in a real time context is useful for live performances (Brossier, Bello & Plumbley 2004).

The decomposition and processing of audio signals into sound objects are emerging fields in music signal processing. Events being the result of segmentation procedures can be stored along with a list of indexes and locations. Resynthesis of the audio consists of juxtaposing the audio segments from the list at their corresponding location. Automated detection, capture and reuse of events, i.e. automated production of databases (Collins 2004) has musical applications in the studio; where a database of sound events can be automatically generated to form source material for composition (Rossignol, Rodet, Soumagne, Collette and Depall 1999; Schwarz 2004; Jehan 2005), and during a live performance; where events are extracted and classified in realtime (Collins 2005).

4.2.2 Classification based on similarity

An ability to assess similarity lies close to the core of cognition. Geometric psychological models have been among the most influential approaches to analyzing similarity and are exemplified by multidimensional scaling (MDS) models. The input to MDS routines may be any measure of subjective similarity between all pairs of entities in a set (e.g. similarity judgments, probabilities of entities being grouped together). The output of an MDS routine is a geometric model of the entities' similarity, with each entity of the set represented as a point in N-dimensional space. The similarity of two entities and is taken to be inversely related to their distance. A Euclidean metric often provides good fits to human similarity judgments when the entities are holistically perceived or the underlying dimensions are psychologically fused, whereas a city-block metric often provides a better fit when entities are clearly divisible into separate dimensions (Goldstone 1994).

A perceptual multidimensional scaling (MDS) of sound was exploited by Grey (1978) who found that traditional monophonic pitched instruments could be represented in a three-dimensional timbre space with axes corresponding roughly to attack quality (temporal envelope), spectral flux (evolution of the spectral distribution over time), and brightness (spectral centroid).

Jehan (2004), employs the labeling of segments in a perceptually meaningful and compact, yet sufficient multidimensional space, in order to estimate their similarities in the timbral sense. Perceptually similar segments should cluster with each other and could therefore hold comparable labels. The similarity between two segments is calculated with a least-square distance measure.

Collins (2004) built a classifier as an experiment in compositional application for event segmentation based on a coarse categorization of captured events. Algorithmic composers running playback of captured events thereby respond to changing timbral events of a live feed from an instrumentalist. The goal of his implementation was classification of incoming sound events as soon as possible, in classes, by means of event feature extraction.

However, neither geometric nor featural models of similarity are well suited for comparing things that are richly structured rather than just being a collection of coordinates or features. Often it is most efficient to represent things hierarchically (parts containing parts) and/or propositionally (relational predicates taking arguments). In such cases, comparing things involves not simply matching features, but determining which elements correspond to, or align with, one another (Goldstone 1999).

4.2.3 Concatenative sound synthesis

Concatenative sound synthesis (Lazier & Cook 2003; Schwarz 2004, 2005, 2006; Aucoutourier & Pachet 2005) is a method of musical sound synthesis with a steady stream of work and publications in recent years. The method was first developed as part of a text-to-speech system, which exploits large databases of speech phonemes in order to reconstruct entire sentences.

Concatenative synthesis methods use a large database of source sounds, segmented into units, and a unit selection algorithm that finds the sequence of units that match best the sound or phrase to be synthesized, called the target. The units can be non-uniform, i.e. they can comprise a sound snippet, an instrument note, up to a whole phrase. The selection is performed according to the descriptors of the units, which are characteristics extracted from the source sounds, or higher-level descriptors attributed to them. The descriptors can be categorical, static, or dynamic. The selected units can then be transformed to fully match the target specification, and are concatenated. However, if the database is sufficiently large, the probability is high that a matching unit will be found, so the need to apply transformations is reduced (Schwarz 2004).

Segmentation can be realized by automatic alignment of music with score (onset-based) or by arbitrary segmenting (grains) an audio signal.

Because concatenative synthesis is aware of the context of the database as well as the target units, it can synthesize natural sounding transitions by selecting units from matching contexts. Information attributed to the source sounds can be exploited for unit selection, which allows high-level control of synthesis, where the units in the database fill in the fine details lacking in the target specification.

Although concatenative sound synthesis proven to be useful for high level instrument synthesis it needs a good model for navigation and efficient search algorithms. Such a model is yet to be presented.

Chapter 5:

Implementations

”In explaining Max patches, the cliché ’One picture is worth a thousand words’ applies. Even a simple patch requires a lengthy textual description if it is assumed that the reader is a novice, as we do here.”
(Curtis Roads, 1999, p.689)

In this chapter we present analytical and compositional tools, implemented in the max/MSP (Puckette 2002) programming environment, which are related to the topic of this thesis.

Those tools, essential for signal-based analytical systems, proven to be useful for onset detection (see section 3.1.1), event segmentation (see section 4.2.1), analysis and classification (see section 4.2.2), and event capture, transformation and concatenation (see section 4.2.4). Additionally, MIDI-based implementations, which form the core of event grouping models (see section 4.1), reflecting principles of Gestalt grouping (see section 2.1, 4.1.1) and the grouping preferences rules of Generative Theory of Tonal Music (see section 4.1.2), are presented.

The author implemented those tools, encapsulated as max/MSP abstractions, as part of his own attempt to integrate analytical with compositional processes, while studying at the Institute of Sonology, in the years 2005-2007. Working versions of those implementations can be found in the CD supporting this thesis, along with example patches.

The current implementations were designed following the criteria ’as simple as possible, whereas retain fidelity’. We hope that the reader will be able to follow despite his/hers programming skills and knowledge of technology related to analytical systems (though some familiarity with max/MSP basics would be necessary). We tried not to complicate matters so the implementations and the corresponding descriptions are ’comprehensible’ by a novice -assuming that (s)he read the sections that are indicated in paragraph 2. We wish that the simplicity of those abstractions would be a reason that students in the future will use them as building blocks in their attempts.

Despite the fact that the current implementations are simple and computationally inexpensive, we do not pretend that are better in any other way than implementations described in the previous chapters of this thesis.

However, the author was surprised to find out, through evaluation tests and comparisons that given their simplicity, the tools yet to be described, compete well-established implementations. This issue is worthy of note since it raises questions regarding the efforts needed to build an analytical module in the trade-off of *’a good model on top of a simple analysis (that) can be more effective than a high quality analysis without any abstraction to deal with its results’* (Pabon 2006).

5.1 max/MSP abstractions

Max is a high level graphical programming environment. Programs are written using graphical objects rather than text. This provides a clear and intuitive way to write programs simply by connecting objects to each other. Max takes care of all the low level programming tasks, thus reduces the need to learn a lot of arcane commands and syntax.

A program written in Max is called a patcher. A patcher is a network of interconnected objects. We call abstractions, objects that were coded entirely in Max.

The Max environment was expanded to include audio data with the introduction of MSP.

5.1.1 onset~

[onset~][11] detects onsets and offsets of notes and sounds in an audio signal, based on amplitude thresholding.

The input audio signal is send through a high-pass filter. The output (signal) of the filter is multiplied with itself. This results to the root-mean-square (RMS) amplitude value of the signal. The RMS amplitude value is sampled every 10ms and is compared with the high detection threshold. If exceeds that threshold [onset~] indicates that an onset was detected. If more than one onsets detected within a certain amount of time, are reported through a separate output.

After an onset is detected [onset~] expects an offset. That means that the RMS amplitude is expected to fall under a low threshold. If no offset is detected up to the moment that a new onset is detected, the new onset is taken as the offset of the previous note or sound. This guarantees that every detected offset corresponds to the preceding onset.

The user has to define the high and low amplitude thresholds that will be used for onset and offset detection respectively, and also the time between successive onsets.

One might change the initial state of the abstraction if required to re-define the coefficients and cut-off frequency of the high-pass filter, or might also change the responsiveness of the envelope follower along with the sample rate.

Moreover the interested Max programmer should be able to expand the functionality of [onset~] by using the statistical objects [peak] and [trough] to turn our abstraction to an adaptive threshold onset detector.

[11] Names in brackets refer to Max objects

5.1.2 event~

[event~] implements onset-based audio signal segmentation. An internal event segmentation process detects sound events, in an audio stream. The detected events are captured in an internal audio buffer. Event indexes and data are store in an internal temporary database.

As soon as the audio signal arrives at the input of [event~], is send through an onset detector. At the same time is recorded and stored temporarily in an audio buffer. A sample-based timer starts a measurement simultaneously with the recording task. Every time an onset is reported, the timer outputs it's current value. This value is stored temporarily, until the next event is detected. Upon the detection of a new event the previous value is subtracted from the timer's current value. Both values are then used to calculate the duration of every previous event. This preserves contiguity in a possible reconstruction of the input audio signal.

The user has to define the high and low amplitude thresholds that will be used for onset and offset detection respectively, and also the time between successive onsets; plus the desirable audio buffer and collection to store the captured events, indexes and data. Those might constitute chunks of a larger database.

It is possible to expand the functionality of [event~] by applying heuristics for salient event detection, thus to capture only target events. Furthermore, is possible to enable [event~] to store data that describe captured events according to parameters used primarily for event detection. These descriptions are useful for automated event classification.

As soon as low-level rhythm analysis centers on event detection (by isolating events in an audio stream and determine their durations), [event~] could find its place as part of a beat tracker implemented in Max.

5.1.3 groups

[groups] is an event grouping abstraction. It calculates and compares intervals in some parameter, while searching for discontinuities in a musical stream. Concurrent discontinuities indicate event group boundaries.

Values arrived from a pitch-to-midi device are temporarily stored and used to calculate intervals. Those intervals are sent through a comparison process that implements the following principle: consider a sequence of four integers i1, i2, i3, i4,
[groups] will assert group boundaries if; the interval between i2 - i3 is greater than that between i1- i2, and that between i3 - i4; the asserted boundaries will be confirmed upon the arrival of i5. If no discontinuity is detected, then the interval values are stored for succeeding comparisons.

Multiple instances of [groups] can be used to compare events in more than one parameter at a time. [groups] operates in real-time, thus is useful for interactive music systems.

Implementing more grouping principles might expand its functionality. This should be as easy as adjusting the expression that is to be evaluated in the core of the abstraction.

5.1.4 concatenate~

[concatenate~] is a concatenative synthesizer. Events, that are stored in a database created by [event~] or other segmentation process, are retrieved, transformed and concatenated, according to instructions asserted externally (defined by the user, or by means of an automated process, e.g. in response to real-time analysis of input audio signal).

The events that are typically retrieved by [concatenate~] can be of various types. The current implementation supports databases that divide events in three classes. Those event classes are labeled after their content as 'pitched', 'percussive' and 'grains'. On the inside of each class events are sorted out according to some parameter: 'proximity' (refers to event position in the segmented audio signal), 'pitch', 'loudness', 'duration' and 'centroid' (spectral centroid; exclusive for 'percussive' and 'grains' classes).

[concatenate~] communicates with a [database] abstraction, and a [control] abstraction. The [database] holds all the necessary information, i.e. captured events, indexes and descriptors. The [control] sends instruction sets to control the internal synthesizer. Those instruction sets include sequences, order manipulations, and various types of transformations based upon the stored descriptors. Those transformations affect event onset, offset, pitch, loudness, duration, inter-onset-interval, amplitude envelope, e.t.c. , and are different for each event. Moreover, the position of each event in a stereo panorama can be defined individually. There are two auxiliary outputs that can be used to further process the output of [concatenate~]. For this purpose data that describe each sounding event are send out while concatenating, so they can be used to control an external processing module or another instance of [concatenate~].
The synthesizer can be also controlled 'externally', e.g. controlled by other instances of [concatenate~], or by a rhythm generator.

When multiple instances of the abstraction are called to operate altogether, is necessary to have some sort of higher-level descriptions (summaries) of the instruction sets. For that reason some abstractions that provide control over 'constrained random numbers' are implemented. Likewise, an abstraction that implements some 'selection principles' is also included. Combinations of those abstractions provide control over, e.g. the density of events at a given time. The same abstractions might be used to express a number of musical gestures.

[concatenate~] is a full-blown implementation. We suggest that the interested Max programmers will take apart the various sub-patches that need to use in their implementations. Though, is also possible that one would like to expand the functionality of [concatenate~] by modifying its internal processes.

5.2 Two example patches

Let us now present two example patches (programs written in Max) and briefly describe their function. Those patches use combinations of the abstractions we described in section 5.1. , next to some other abstractions that we didn't had the opportunity to describe before. Still, all the necessary abstractions are included in the cd that supports this thesis.

Both examples include analytical and compositional processes. Those processes are intentionally and plainly divided, as soon as the following patches are merely examples that demonstrate the functionality of our abstractions.

5.2.1 EventCutUp

'EventCutUp' constitutes a signal network that captures events of an audio stream, transforms and concatenates them. The captured events and descriptions are stored in a database that can be exported and used by other programs. The user controls the concatenation process by giving instructions in real-time. Alternatively, a rhythm generator controls the process, while the user is able to adjust its performance parameters. The output of this process is recorded and saved on the hard disk.

The patch is divided into analytical and compositional processes. The input audio signal is segmented based on three segmentation schemes: onset detection using pitch, onset detection using loudness, and fixed length segmentation (10-100ms). This results to corresponding classes in the database. A database can be exported and imported. That means that we do not have to re-analyze soundfiles.

Amongst the compositional arsenal of the patch a concatenative synthesizer can be found. All the processes are automated so the user has to only be concerned with providing supplies to a selection process and choose between various selection principles. This can be done through a users interface encapsulated in the 'control' sub-patch. Those supplies are actually the legal boundaries of a constrained random number generator, and the principle (random, random with repetition check, e.t.c) to use to select each number within these boundaries. Later implementations the user the opportunity to use tendency masks, for changing over time the boundaries of a constrained random generator.

The interested composer might use this patch to generate material up to the meso-level of a musical composition.

5.2.2 EventGroup

'EventGroup' is a MIDI based network that divides a stream of notes into groups in real-time and responds, with an interpretation of the input and by generating new material.

The analytical processes of this patch use an implementation based on some of the Grouping Preference Rules of Lerdahl and Jackendoff's Generative Theory of Tonal Music as described in section 4.1.2.

The compositional processes of 'EventGroup' involve an artificial 'player' that uses the results of analysis to play a free interpretation of the input.

The 'listener' agent of this patch is able to 'comprehend' any MIDI input, while the 'player' agent has always something to say.

As soon as all the processes are real-time, and fully automated, 'EventGroup' could be used as part of an interactive music system.

Chapter 6:

Discussion

”Every golden apple has its own golden worm”
(Robert Shea and Robert Anton Wilson, 1975, p. 730)

We need not concern ourselves with accusing analytical systems of not been able to do what were never designed to do.

What we consider valuable at this moment is to reassess the suggestion that:

’analytical processes could be also suitable for composition’.

6.1 Argumentum

Proponents of the suitability of analytical processes for music composition argue that the rules accumulated by a uniform analysis could provide also control over the way a listener will parse a musical structure. But it could also be argued that those rules are shallow and formative. Simply because certain principles, which follow not necessarily correct analogies between vision and hearing, or between linguistics and music, apply to some number of examples is no guarantee that a listener will hear musical structures as has been asserted by an exhausting though not exhaustive analysis. Musical parsing and musical affect convey cognitive categories for which little theory yet exists. If there is a way to control formal structures as to become ’perceptually effective’ (Tenney and Polansky 1980) there is not an obligation to do so by means of explicit analysis. The composer, being a listener himself, can use his ears to evaluate his music.

It would be a mistake to think music theory in general as pursuing the same goals as music cognition. Even if some work in music theory might be regarded as introspectionists’ cognitive science (Temperely 2001). However, music theory and music cognition share some aspects that distinguish them from music composition. Both are inherently normative, and reflect a codification of past achievements. Regarding codification, music theory includes a collection of procedures that describe musical discourse, while music cognition attempts to describe the associated mental representations. In any case, previously acquired knowledge can influence and hamper the uptake of new information (proactive interference) giving rise to limited biased discourses and prejudices. Music composition is creative and expands the theory (Berg 1996).

Rowe (2001) makes this observation regarding music analysis: *’the result of an analysis is a written, rational document that may be examined for formal constructions amenable to implementation in a computer program. It is not, however, the inverse transform of an act of composition. At the end of an analysis we are not back at the composer’s thoughts*’. Music analysis should not be mistaken for reverse engineering of music composition.

Amongst the fundamental differences between analytical and compositional processes is that the latter might involve any arbitrary starting point, or any arbitrary manipulation of structure, while the former is called to reason using a concrete set of principles.

Music can be parsed into high-level structures in innumerable ways (Roads 1999). Analysis is the means to reveal, mere to suggest, structures in several levels, as to examine their content and relation with other structures within a given context, by deduction from music theory and other formative approaches, i.e. music perception and cognition. In contrast, an exclusive aspect of music composition is to combine those structures within a chosen context. To define this context is a compositional decision by itself. Particularly, composing with computers is a broad area where the composer has the opportunity to define sole, subsequently novel, contexts.

On the other hand, is a reasonable hypothesis if we consider the incremental improvements in music analysis/synthesis systems, that creating music (synthesis) *'could share the same knowledge as acquired from a uniform analysis procedure based on perceptual listening and learning*' (Jehan 2005). This might be the case when a system is merely focus on creating 'mash-up' music (the practice of making new music out of previously existing recordings). The results might be intriguing, but not necessarily compelling aesthetically. From a technological point of view, 'signal-based unbiased automation' is indeed an achievement. Then again, example-based systems lack originality.

Rules for creating music obtained by analysis are merely 'cognitive artifacts' (a category of processes that produce cognitive effects by bringing functional skills into coordination with various kinds of structure). The utility of a cognitive artifact depends on other processes that create the conditions of its use. In the case of music the processes that create those conditions are by default compositional.

One of the principal findings of studies of cognition and learning is that people make opportunistic use of structure. Equally, the context in which a structure is placed will ascertain its interpretation. Together with the fact that the nature of cognitive processing is uniquely determined within its context, and that it cannot be studied in isolation without destroying its defining properties, leads to the implication that an analytical system can be valid only within the context from which was deduced. Using an analytical system in a different context, or will force novel constructs to match its assumptions, or will be rendered useless. Nevertheless, the merit of a tool suitable for composition is to be general and be able to be implemented in a wide range of musical applications.

Roads (1999) put in writing: *'as we listen, part of us drinks in the sensual experience of sound, while another part is constantly setting up expectations, and in so doing, constructing hypotheses of musical processes*'. In fact, musical schemata emerge (when was the last time you heard a 'new' sound?). Those control structures in the human brain that are sensitive to some frequently occurring pattern, either in the environment, in ourselves, or in how the two interact (Bregman, 1990) really affect the way we listen to music, therefore affecting analytical and compositional processes. However, if schema acquisition is rapid (Alty 2002), then the composer has the opportunity to build up expectations throughout the course of a single piece (or a series of pieces). In contrast, it takes years of research to build analytical systems able to induct the simplest musical structure.

It has been argued that: *'most attempts to build music-analysis systems have tried hard to respect the conventional wisdom about the structure of music. However, this model of music—based on notes grouped into rhythms, chords and harmonic progressions—is really only applicable to a restricted class of listeners; there is strong evidence that non-musicians do not hear music in these terms. As a result, attempts to directly apply ideas from music theory and statistical signal processing have not yet led to successful musical multimedia systems. Today's computer systems are not capable of understanding music at the level of an average five-year-old; they cannot recognize a melody in a polyphonic recording or understand a song on a children's television program. We believe that to build robust and broadly useful musical systems, we must discard entrenched ideas about what it means to listen to music and start again*.' (Martin, Scheirer and Vercoe 1998, p.1). Even if those problems have to do with the current state of the art, so it is possible that in the future it will be solved, *when dealing with music signals and extracting perceptual information, there is necessarily a fair amount of ambiguity and imprecision in the estimated data, not only due to the analysis technique, but also to the inherent fuzziness of the perceptual information* (Jehan 2005).

Nevertheless, analytical systems in recent years became highly sophisticated and complicated in the trade-off of *'a good model on top of a simple analysis (that) can be more effective than a high quality analysis without any abstraction to deal with its results'* (Pabon 2006).

6.2 Conclusions

Analytical and compositional processes are fundamentally different.

Analytical systems rely on formalizations of knowledge as acquired by means of music theory and music cognition. Those are deductive models that might help us automate certain aspects of a computer-assisted composition system.

Composition, being a creative process, can articulate an infinity of musical constructs. In contrary, the resultant structures can only be analyzed with finite resources. Analytical processes and the analogous systems, being implemented music theory and cognition, are inherently normative.

If there is anything interesting in composing, with or without computers, is the opportunity to deal with abstract concepts while manipulating concrete material. In particular, computer-based compositional systems provide extended control over sound properties and aspects of musical structure presenting the opportunity to merge those two into new musical forms.

6.3 Suggestions

Not every musical culture produces music theories, but every member of a culture that listens to music exercises a cognitive capacity. Therefore, music cognition could improve the efficiency of analytical systems towards the design of models grounded more in the way that humans listen to music whatever their knowledge of music theory is.

We consent that a good model on top of a simple analysis can be more effective than a high quality analysis without any abstraction to deal with its results. We anticipate the design and implementation of compositional algorithms that are capable to use such an abstraction to change the parameters that govern their interactions with their musical environment.

Appendix:

max/MSP abstractions and patches

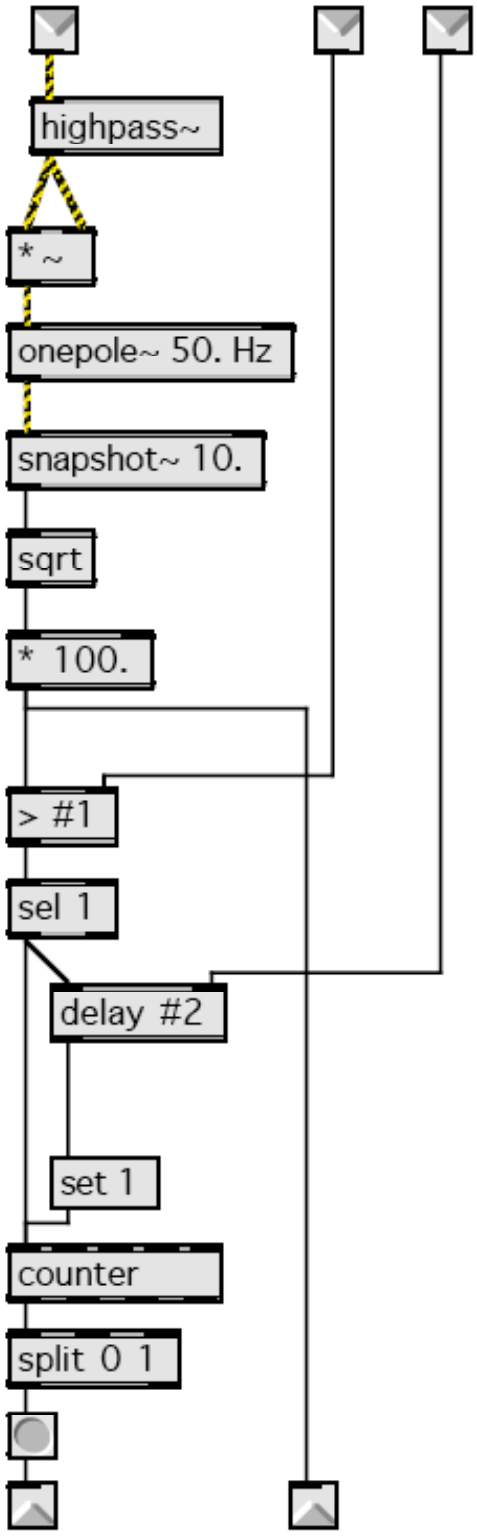


figure 1. onset~

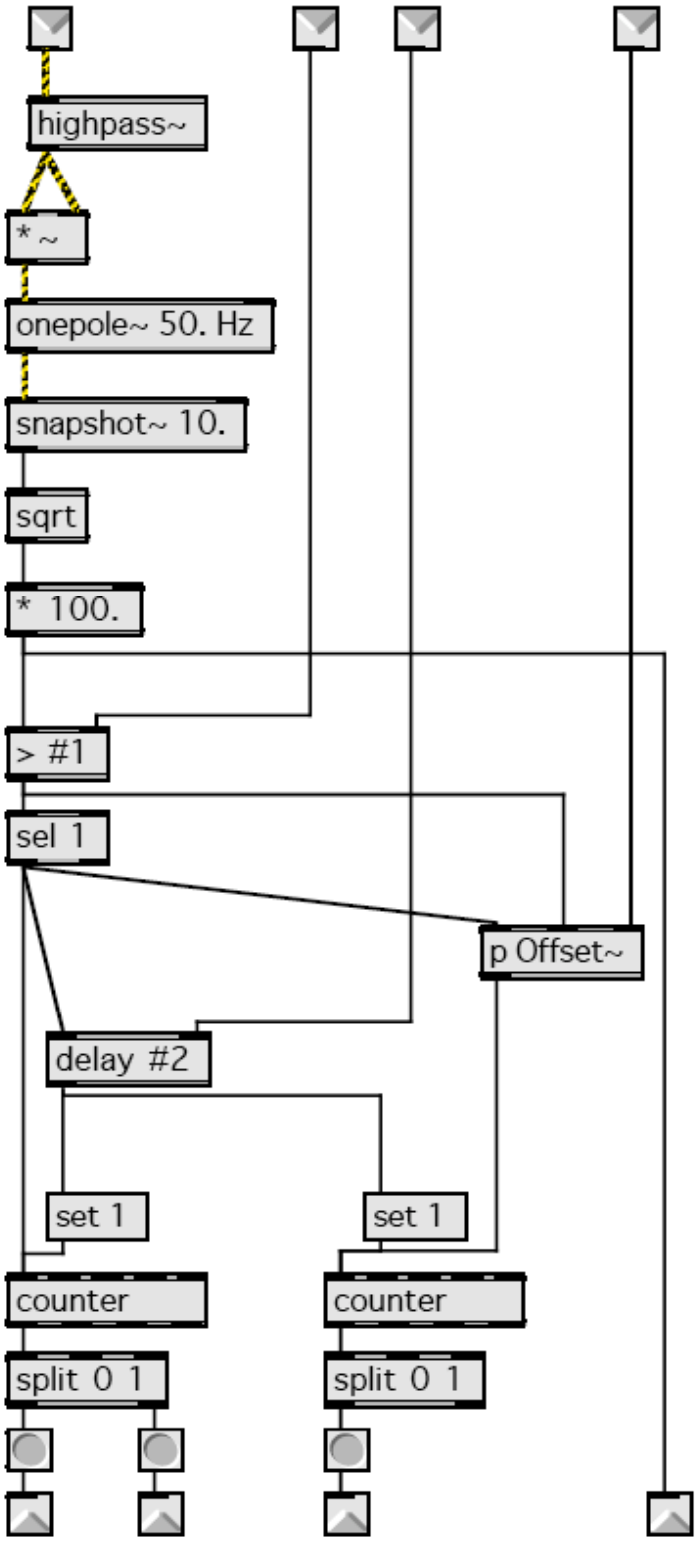


figure 2. onset2~

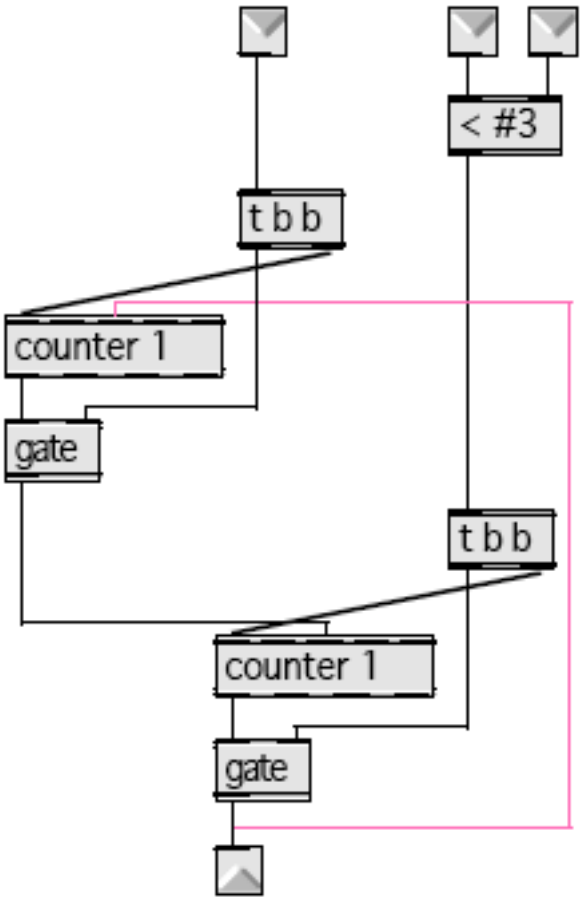


figure 3. Offset~

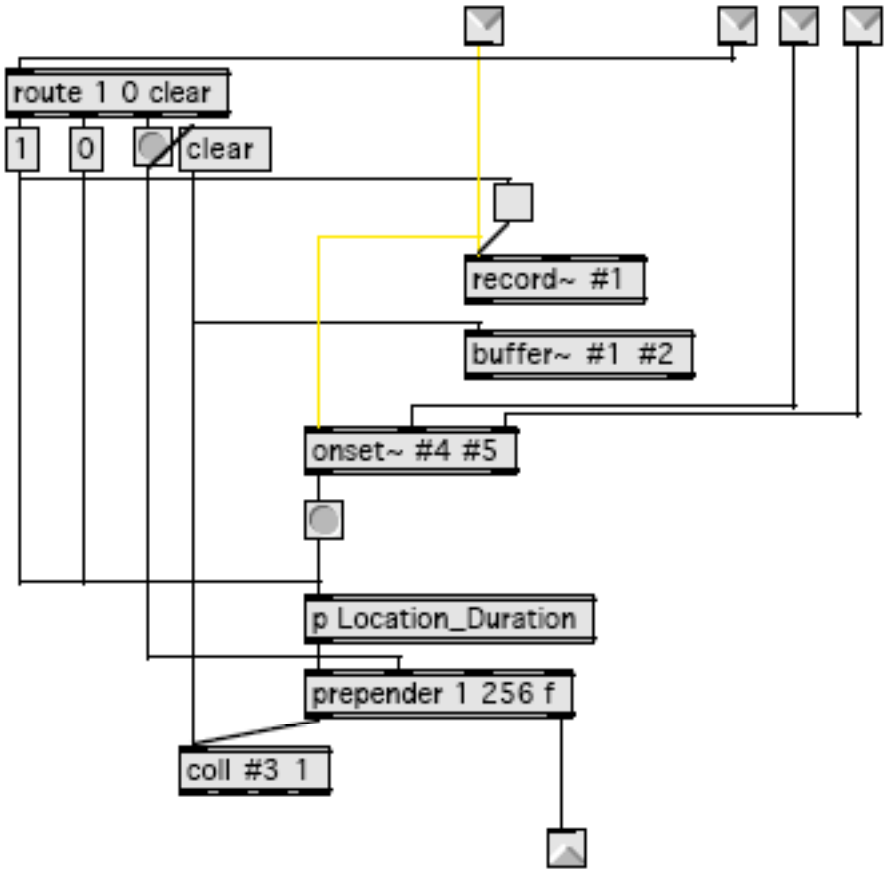


figure 4. event~

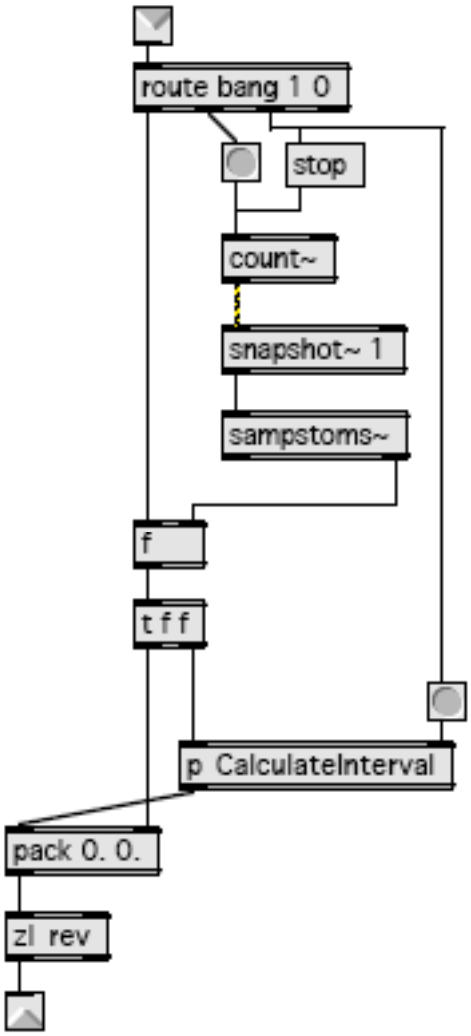


figure 5. Location_Duration

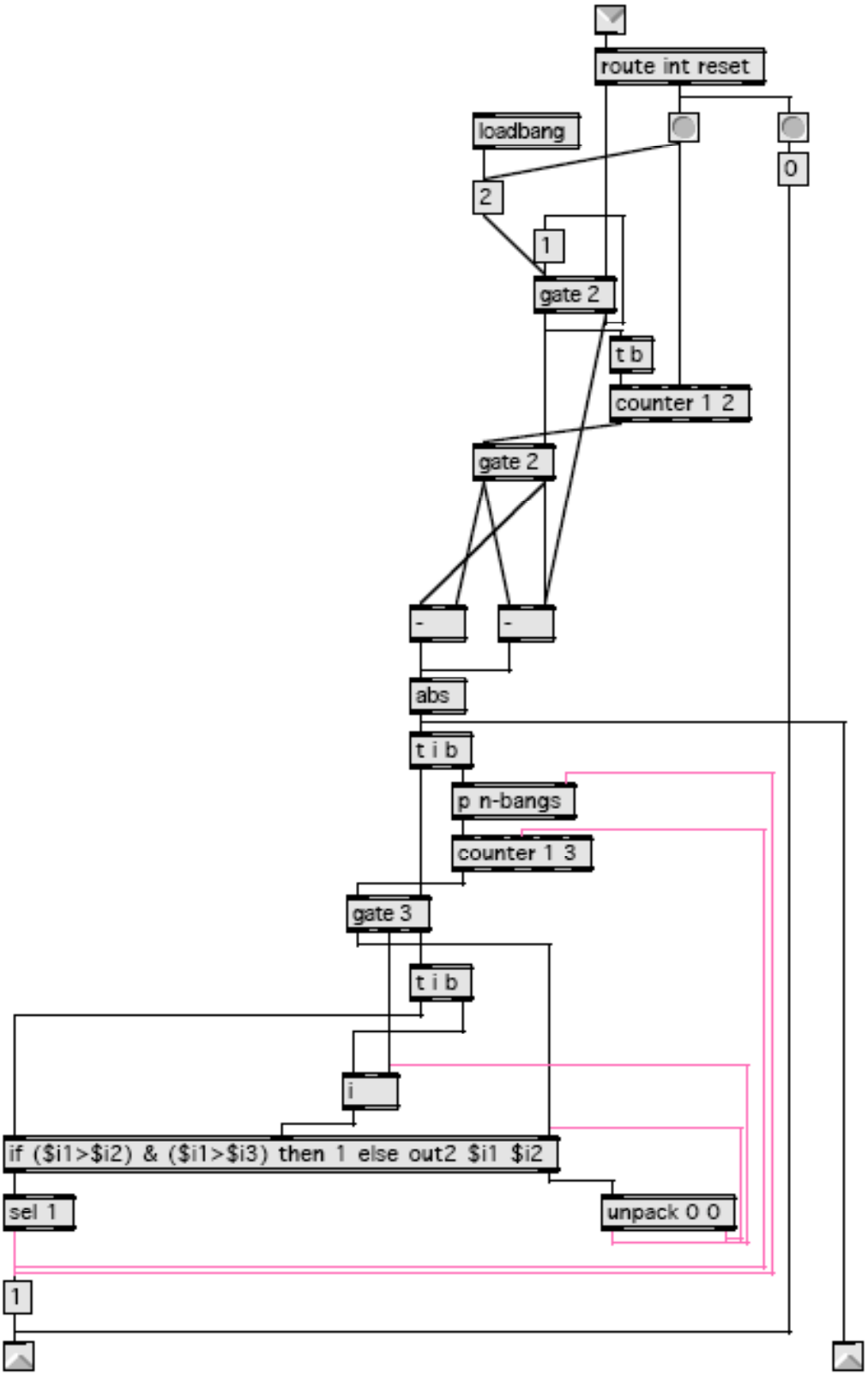


figure 6. groups

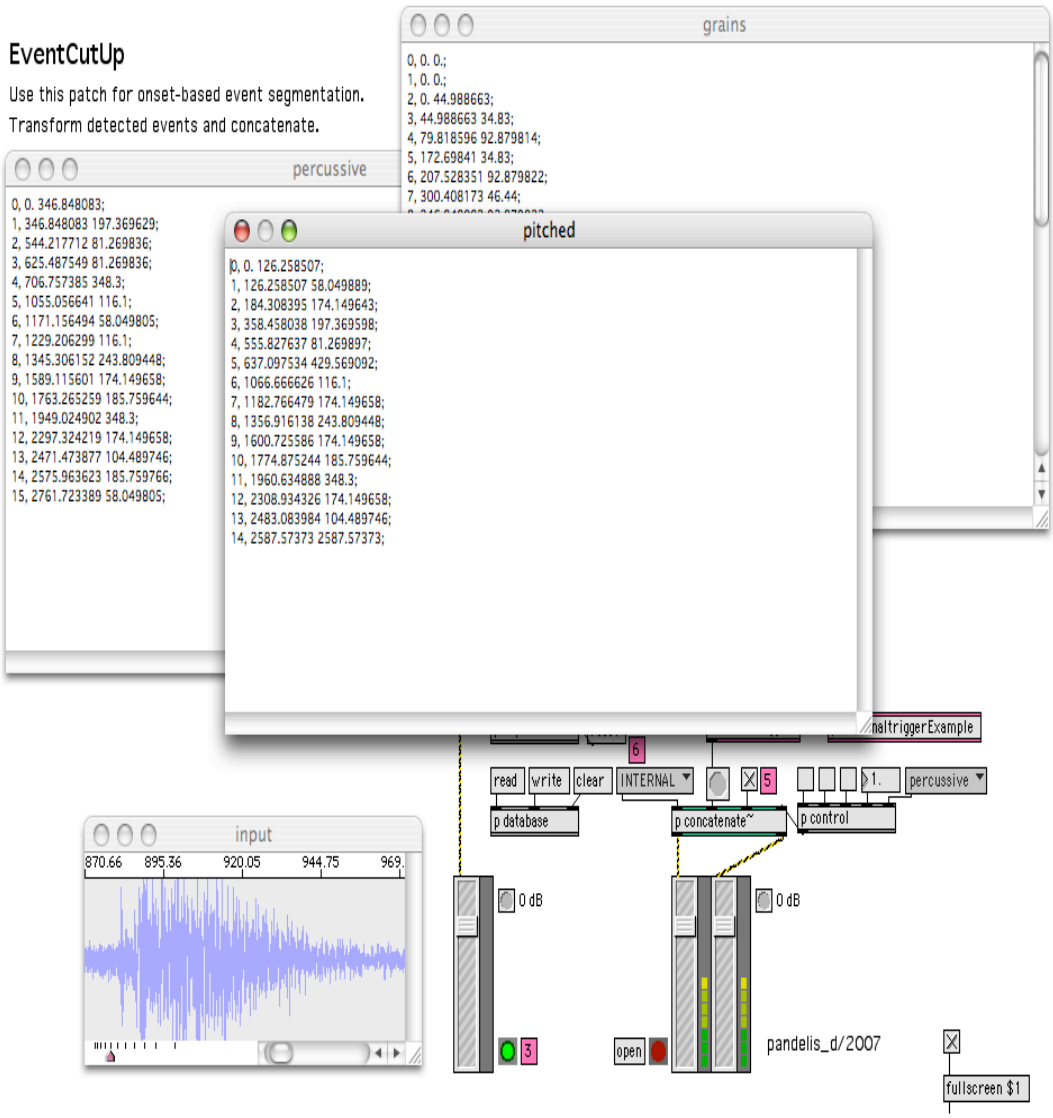


figure 7. EventCutUp

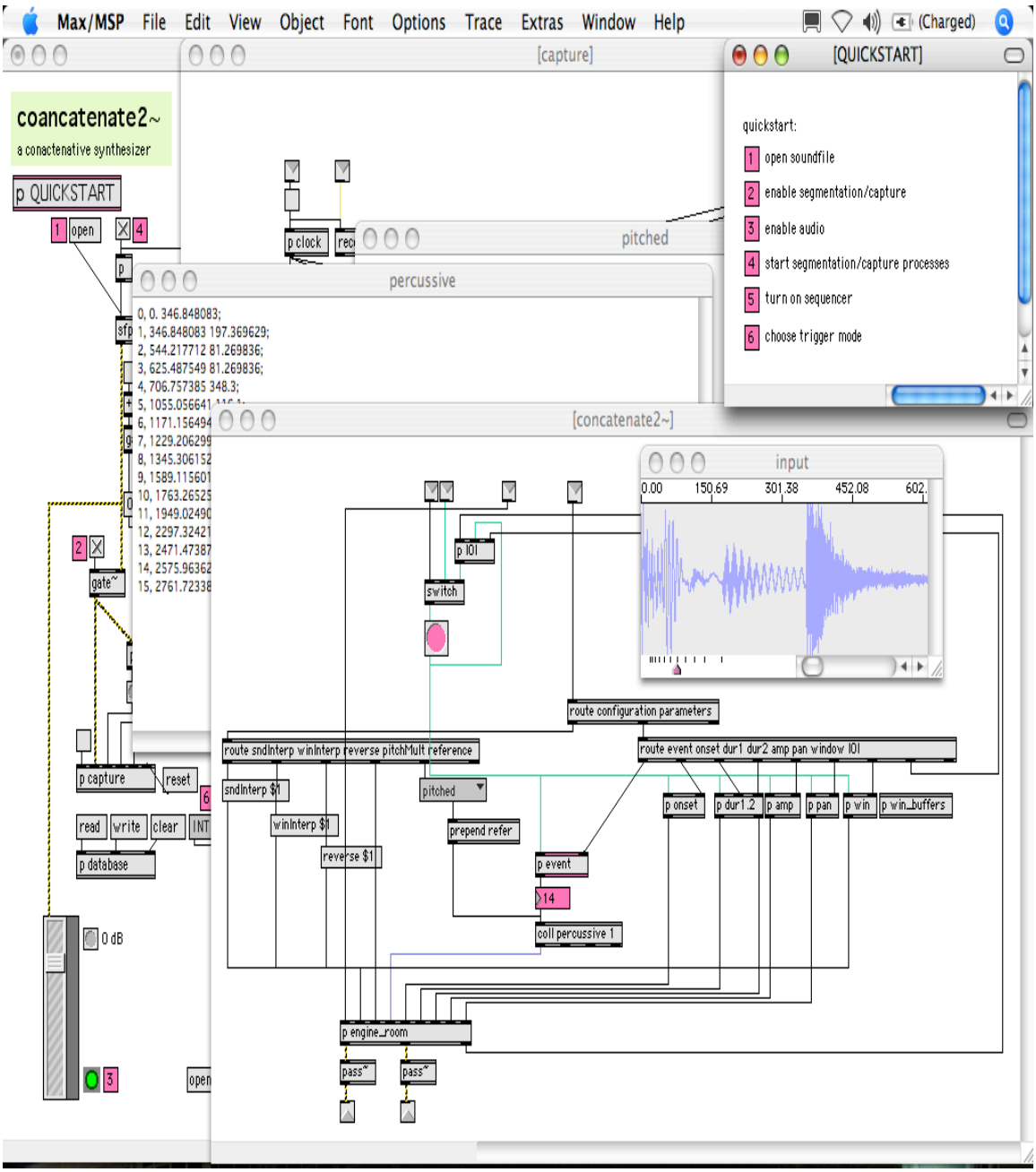


figure 8. concatenate~

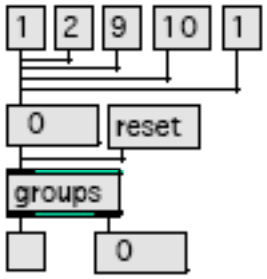
groups

calculates and compares intervals. Useful for event grouping

inlet: values, reset

outlet 1: group boundaries (end of a group = beginning of a new one)

outlet 1: last interval



Calculates interval between last and current value

Compares intervals according to the following principle:

Consider a sequence of four integers i1i2i3i4:

intervals will output 1 (right outlet) if

the interval i2 - i3 is greater than that from i1- i2 and that from i3 - i4

note: calculations complete upon the arrival of i5

pandelis_d/2007

figure 9. groups.help

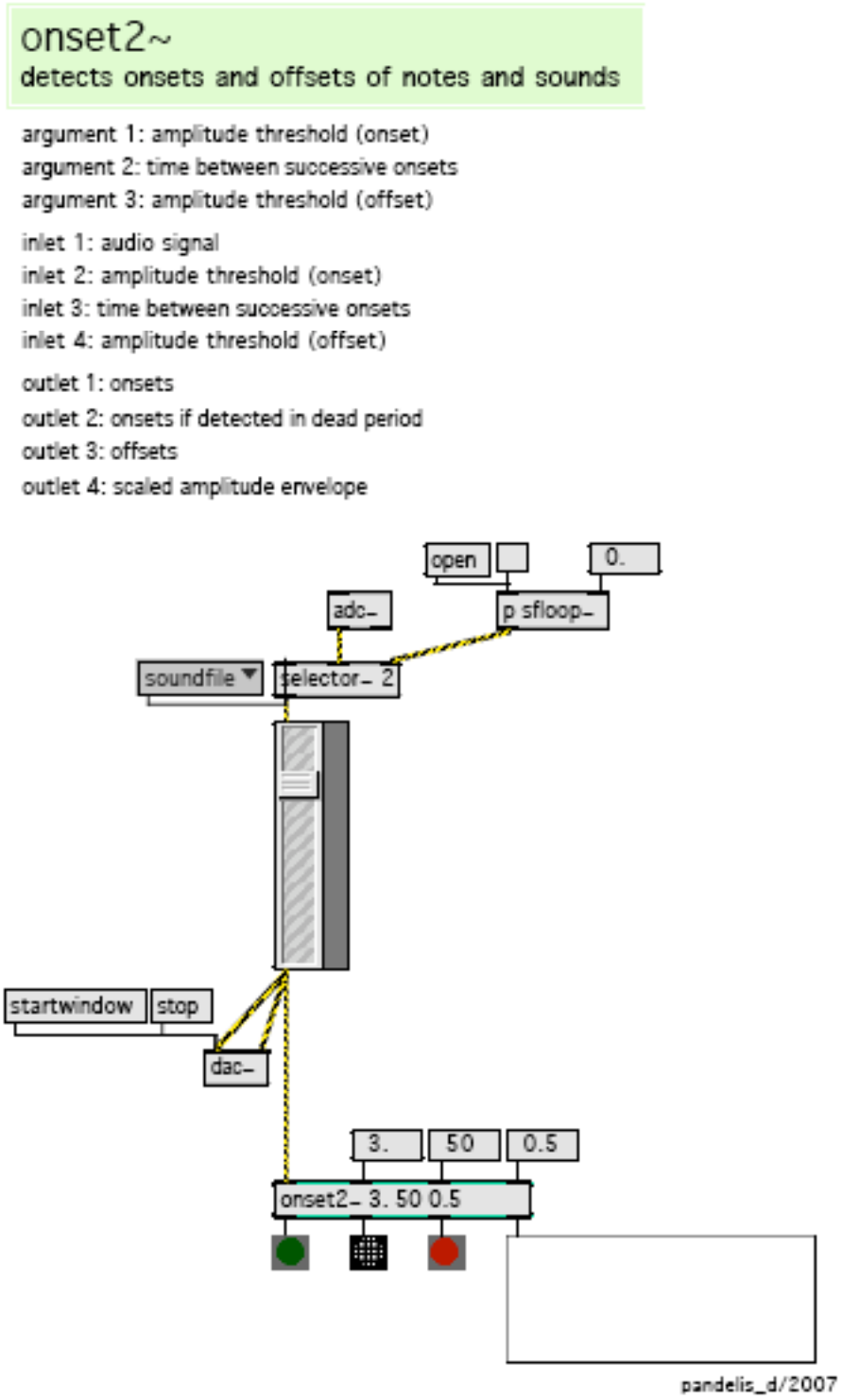


figure 10. onset2~.help

References:

Alty, J. (2002). ”Engineering for the mind: Cognitive science and Musical Composition”. Journal of new music research, vol.31, no.3, pp. 249-255.

Aucoutourier, J. -J. and Pachet, F. (2005). "Ringomatic: A real-time interactive drummer using constraint-satisfaction and drum sound descriptors”. In Proceedings of the International Conference on Music Information Retrieval. London.

Aucoutourier, J. -J. and Pachet, F. (2006). "Jamming With Plunderphonics: Interactive Concatenative Synthesis Of Music". Journal of New Music Research.

Berg P. (1996). ”Abstracting the Future: The Search for Musical Constructs”. Computer Music Journal, Vol. 20, Issue 3.

Bod R. (2000). ”A Memory-Based Model for Music Analysis: Challenging the Gestalt Principles”. University of Amsterdam, Spuistraat 134, 1012 VB Amsterdam, NL.

Bregman A. (1990). "Auditory Scene Analysis: The Perceptual Organization of Sound." Cambridge, MA: MIT Press.

Brossier P., Bello J. and Plumbley M. (2004) "Real-time temporal segmentation of note objects in music signals". In Proc. Int. Computer Music Conference.

Cambouropoulos, E. (1998) ”Towards a General Computational Theory of Musical Structure”. Ph.D. The University of Edinburgh Faculty of Music and Department of Artificial Intelligence.

Carroll, L. (1865-1871/ 1944). ”Alice’s adventures in wonderland and Through the looking glass”. New York: Macmillan.

Clarke E. F. (1999). ”Rhythm and timing in music”. In D. Deutsch (Ed.), The psychology of music. (2nd ed., pp. 473-500). New York: Academic Press.

Coldstone (1999). ”Similarity”. In The MIT encyclopedia of the cognitive sciences. Ed. Robert A. Wilson and Frank C. Keil. Cambridge, MA. MIT Press.

Collins, N. (2004). " On Onsets On-the-fly: Real-time Event Segmentation and Categorization as a Compositional Effect”. In Sound and Music Computing (SMC04), pages 219–24, IRCAM, Paris.

Collins, N. (2005a). "A comparison of sound onset detection algorithms with emphasis on psychoacoustically motivated detection functions". In AES Convention 118, Barcelona.

Collins, N. (2005b). "An Automated Event Analysis System with Compositional Applications". Proceedings of the International Computer Music Conference, Barcelona.

Collins, N. (2007). "Towards Autonomous Agents for Live Computer Music: Realtime Machine Listening and Interactive Music Systems". PhD Dissertation. University of Cambridge.

Cope, D. (1991). "Computers and Musical Style". Oxford University Press, New York.

Cope, D. (1992). "Computer Modeling of Musical Intelligence in EMI". Computer Music Journal, 16(2): 69-83.

Cope, D. (1993). "A Computer Model of Music Composition". In Machine Models of Music, S.M. Schwanauer and D.A. Levitt (eds), The MIT Press, Cambridge (Ma).

Deliege, I. (1987). "Grouping Conditions in Listening to Music: An Approach to Lerdhal and Jackendoff's grouping preferences rules". Music Perception, 4:325–360.

Desain, P. and Honing, H. (1994). "Can music cognition benefit from computer music research? From foot-tapper systems to beat induction models". In Proceedings of the ICMPC. 397-398. Liege: ESCOM.

Deutsch, D. and Feroe j. (1981). "The Internal Representation of Pitch Sequences in Tonal Music". Psychological Review, 88, 503-522.

Deutsch, D. (1999). "Grouping mechanisms in music". In The psychology of music. 2nd Ed. Ed. D. Deutsch. London: Academic Press.

Dietterich, T. (1999). "Machine Learning". In The MIT encyclopedia of the cognitive sciences. Ed. Robert A. Wilson and Frank C. Keil. Cambridge, MA. MIT Press.

Dowling, W. and Harwood D. (1986)."Music cognition". New York: Academic Press.

Essl, K. (2004). "RTC-lib: Real Time Composition Library for Max". http://www.essl.at/works/rtc.html

Goldstone, R. L. (1994). "An Efficient Method for Obtaining Similarity Data."

Behavior Research Methods, Instruments, & Computers 26:381–386.

Grey, J. (1978). “Timbre discrimination in musical patterns,” Journal of the Acoustical Society of America, vol. 64, pp. 467–472.

Jehan, T. (2004). ”Event-synchronous music analysis/synthesis”. In Proc. Digital Audio Effects Workshop (DAFx), Naples, Italy.

Jehan, T. (2005). ”Creating music by listening”. PhD Thesis. Program in Media Arts and Sciences, School of Architecture and Planning, MIT.

Kasabov, N. (1996). ”Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering”. The MIT Press Cambridge, Massachusetts London, England.

Klapuri, A. (1999). ”Sound onset detection by applying psychoacoustic knowledge”. In Proc. IEEE Int. Conf. Acoust., Speech, and Signal Proc. (ICASSP), pages 3089-92.

Lazier, A. and Cook, P.(2003). ”MoSievius: Feature driven interactive audio mosaicing”. In Proc. of the 6th Int. Conference on Digital Audio Effects (DAFx-03).

Lerdahl, F. and Jackendoff, R. (1983). ”A generative theory of tonal music”. Cambridge, MA: The MIT Press.

Lerdhal, F. (1987) ”Timbral hierarchies”. Contemporary Music Review, 2:135 - 160.

Martin K., Scheirer E. and Vercoe B. (1998).“Musical content analysis through models of audition”, in Proc.ACM Multimedia Workshop on Content-Based Processing of Music, Bristol, UK.

Minsky, M. (1988) ”The Society of Mind”. Touchstone Editions, New York.

Pabon, P. (2006) quoted in: ”A Real-Time Timbre Tracking Model Based on Similarity”. Satoshi Shiraishi. MA Thesis. Institute of Sonology Royal Conservatory, The Hague, 2006.

Palmer (1999). ”Gestalt Perception”. In The MIT encyclopedia of the cognitive sciences. Ed. Robert A. Wilson and Frank C. Keil. Cambridge, MA. MIT Press.

Pennycook, B., Stammen, D. and Reynolds, D. (1993). Toward a computer model of a jazz improviser. In Proceedings of the International Computer Music Conference. San Francisco: International Computer Music Association.

Puckette, M. (2002). ”Max at seventeen”. Computer Music Journal, 26(4): 31–43.

Roads, C. (1999). ”Computer music tutorial”. Cambridge, Mass: MIT Press.

Rossignol S., Rodet X., Soumagne J., Collette J. –L. and Depalle P. (1999). ”Automatic characterisation of musical signals: Feature extraction and temporal segmentation”. Journal of New Music Research, 28(4): 281–95.

Rowe, R. (1993). ”Interactive Music Systems”. MIT Press, Cambs, MA.

Rowe, R. (1996). ”Incrementally Improving Interactive Music Systems”. Contemporary Music Review, Vol. 13, Part 2, pp. 47-62.

Rowe, R. (1999). ”The Aesthetics of Interactive Music Systems”, Contemporary Music Review, Vol. 18, Part 3, pp. 83-87

Rowe, R. (2001). ”Machine Musicianship”. MIT Press, Cambs, MA.

Rumelhart D. (1989) ”Mind Design II : Philosophy, Psychology, Artificial Intelligence”. ed. John Haugeland. The MIT Press.

Scheirer E. (1998). “Tempo and beat analysis of acoustic musical signals”. J.Acoust.Soc.Am, vol. 103, no. 1, pp. 588,601.

Schwarz, D. (2004). "Data-Driven Concatenative Sound Synthesis" . PhD thesis, Universit´e Paris.

Schwarz, D. (2005). "Current research in concatenative sound synthesis". Proceedings of the International Computer Music Conference, Barcelona.

Schwarz D., Beller G., Verbrugghe B., Britton S. (2006). "Real-time corpus-based concatenative synthesis with catart". Expanded version 1.1 of submission to the 9th Int. Conference on Digital Audio Effects (DAFx-06), Montreal, Canada.

Shea, R. Wilson, R. A. (1975). ”The Illuminatus! Trilogy”. Constable & Robinson, London, UK.

Smith, L. (1994). ”Sound segmentation using onsets and offsets”. Journal of New Music Research, 23:11–23.

Snyder, B. (2000). ‘’Music and Memory: an Introduction”. MIT Press, Cambridge, MA.

Tempelaars, S. (1996). ”Signal Processing, Speech and Music”. Swets & Zeitlinger B.V., Lisse.

Temperely, D. (2001). ”The cognition of Basic Musical Structures”. MIT press.

Tenney, J. (1961). ”Meta+Hodos”. Frog Peak Music, Oakland (Ca).

Tenney, J. and Polansky L. (1980). ”Temporal Gestalt Perception In Music”. Journal of Music Theory, 24: 205-241.

Todd, N. (1994). ”The auditory “primal sketch”: A multi-scale model of rhythmic grouping”. Journal of New Music Research 23(1):25–70.

Toiviainen, P. and Snyder, J. (2000). ”The time-course of pulse sensation: Dynamics of beat induction”. In C. Woods, G. Luck, R. Brochard, F. Seddon, & J. A. Sloboda (Eds.), Proceedings of the Sixth International Conference on Music Perception and Cognition. Keele: Keele University.

Tzanetakis G. and Cook P. (1999). "Multifeature audio segmentation for browsing and annotation", Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, New Paltz, New York, 17-20.

Wertheimer, M. (1923). Untersuchungen zur Lehre von der Gestalt. Psychologische Forschung 4, 301-350.

West K. and Cox S. (2005). "Finding an optimal segmentation for audio genre classification", Queen Mary, University of London.