

---

**Bob L. Sturm**

Department of Electrical and Computer  
Engineering and Media Arts and Technology  
Program  
University of California, Santa Barbara  
Santa Barbara, California 93106 USA  
b.sturm@mat.ucsb.edu

# Adaptive Concatenative Sound Synthesis and Its Application to Micromontage Composition

Adaptive concatenative sound synthesis (ACSS) is a recent technique for generating and transforming digital sound. Variations of sounds are synthesized from short segments of others in the manner of collage based on a measure of similarity. In electroacoustic music, this has been done by manually locating, categorizing, arranging, and splicing analog tape or digital samples—a style termed *micromontage* (Roads 2001, pp. 182–187). This is akin to performing granular synthesis by hand.

Instead, ACSS provides an intuitive way to automate and control this procedure, freeing time for experimenting and composing with this flexible sound-synthesis technique. As an automation of micromontage methods informed by signal processing and extended to databases of any size, ACSS provides new paths for the art and science of sound collage. Sound synthesis and design—the general organization of sound objects—can develop in directions that are manually prohibitive. Additionally, ACSS provides a creative interface to large databases of sound, where the “query-by-example” paradigm becomes “synthesize-by-sound-example.”

Through several illustrative examples, including compositions exploring aspects of the algorithm, this article attempts to demonstrate the potential of ACSS for these applications. A brief history of micromontage and the concepts behind ACSS are presented. Several implementations are reviewed, and a new one is demonstrated and compared to these. Finally, its application to composing in the style of micromontage is explored and analyzed. In the end, it is argued that ACSS provides a flexible, intuitive, and natural way to shape sound.

## A Brief History of Micromontage

The notion of creating sound by concatenation is not far removed from electroacoustic tape works by composers John Cage, Iannis Xenakis, Bernard Parmegiani, and James Tenney; the more recent digital “microsonic” works of Horacio Vaggione and Curtis Roads; the “plunderphonics” of John Oswald; or the “hyperrealism” of Noah Creshevsky. These composers have manually performed the laborious process of selecting and fusing together even the shortest segments of recorded sound—so short that their results exist more in the realm of micromontage than of *musique concrète*.

In 1952, John Cage composed *Williams Mix* using chance operations. The work features a 192-page score prescribing the arrangement and splicing of about 600 small pieces of magnetic audiotape (Kostelanetz 1970, pp. 109–111). Each sound to be used comes from one of six categories, such as city sounds, country sounds, electronic sounds, etc. It took Cage and at least five others nine months to record and collect sounds and splice pieces of tape to create a four-minute realization (Kostelanetz 1970, pp. 130)—one of the first electroacoustic works composed in the United States.

Similarly, in the late 1950s, Iannis Xenakis spliced together hundreds of short magnetic tape segments to create *Concrèt PH* and *Analogique B* (Xenakis 1992, pp. 103–109; Roads 2001, pp. 64–66)—though it is believed Bernard Parmegiani assembled the tape for the latter (Roads 2005). Mr. Parmegiani used the same technique to construct parts of his 1967 electroacoustic masterpiece *Capture Éphémère* (Parmegiani 2002), which has moments resembling the characteristic sparkling of *Concrèt PH*.

Extensions of this technique into the digital domain can be found in the work of Horacio Vaggione and Curtis Roads. In *Schall* (Vaggione 1995), com-

---

posed in 1995, Horacio Vaggione transforms and arranges thousands of segments of piano sounds to create a variety of textures and themes (Roads 2001, pp. 312–316). Software he has recently helped develop for this purpose provides an intuitive and powerful interface to working with sound at numerous time scales (Vaggione 2005). Within this application, a user can isolate, organize, arrange, and group sound segments, as well as perform transformations on them. Essentially, it is an interface for controlling and organizing the “granulation” of sounds.

Curtis Roads’s work *Half-life* (Roads 2004), composed in 1998, uses “atoms” of material generated by pulsar synthesis (Roads 2001, pp. 137–157). Mr. Roads writes in the liner notes to the recording that “the strategy was to generate the sounds, to study and classify the sounds, to pick, choose, and transform the sounds, and finally to connect and layer the sounds. . . . *Half-life* was honed in microscopic detail, on a particle-by-particle basis.” Like Mr. Vaggione, Mr. Roads uses custom designed software to generate material.

Composer Noah Creshevsky applies the idea of micromontage to create what he calls “hyperrealism,” which he describes as “an electroacoustic music language constructed from sounds that are found in our shared environment . . . handled in ways that are somehow exaggerated or excessive” (Creshevsky 2003, liner notes). Fundamental to his style are superhuman performances and the incorporation of sounds from around the world to create a “sonic bonanza” (Creshevsky 2005). He takes care in finding and extracting each sample he uses to “eliminate the tiny telltale signs that mark their origins” (Creshevsky 2001). In his work *Borrowed Time* (Creshevsky 1995), composed in 1992, he combines short fragments of recorded vocal music from compositions dating between the 12th and 20th centuries. The composer writes that “[e]ach sample consists of a solitary musical event, the duration of which is generally shorter than one second. I generated melodies, phrases, and harmonic progressions by arranging and rearranging tiny bits of embryonic musical matter” (Creshevsky 2001, p. 91).

Other composers such as James Tenney and John

Oswald make use of samples in much the same way, but with sound sources pregnant with cultural significance. James Tenney uses the same tape-splicing techniques as Iannis Xenakis and Bernard Parmegiani in his 1961 composition *Collage #1 (Blue Suede)* (Tenney 1992). Here, he humorously recomposes sound material from Elvis Presley’s rendition of Carl Perkins’s song “Blue Suede Shoes.” John Oswald’s work in this vein is what he calls “plunderphonics,” in which tiny but identifiable pieces of musical culture are used to compose new work rich in popular signifiers (Holm-Hudson 1997; Oswald 2001, 2006). His 19-minute work *Plexure* (1993) is an incredible assemblage combining over 2,000 fragments of popular music from 1982 to 1992. Some segments come and go just as they can be identified, and others hang around to provide motivic material. Interestingly these short samples are recognized more by their timbre than higher-level content such as melody or context (Holm-Hudson 1997).

Compared with the manual labor performed by these composers, the advantages of automating the procedure of selecting and concatenating small sound segments are clear and exciting: the labor in assembling by hand a desired sound from potentially hundreds of segments is instead spent on experimentation, fine tuning, and composing with the results. The “library” from which sounds are selected is limited only by available disk space; for example, one could use the complete recorded works of Beethoven. The idea of automatically transmuting timbre and form is alluring. The concepts behind ACSS are now presented, followed by a review of several implementations.

### **Adaptive Concatenative Sound Synthesis**

For over two decades, concatenative sound synthesis (CSS) approaches have been popular for realistic text-to-speech synthesis (Hunt and Black 1996), herein termed concatenative text-to-speech (CTTS). By using a database of real speech waveforms segmented into “units”—such as the “j” in “jam,” “a” in “bat,” and the “z” sound in “is”—these can be

---

concatenated to synthesize the spoken word “jazz.” Selecting or modifying waveforms based on context, prosody, and inflection increases the realism. An alternative approach for speech synthesis uses a parametric model (Klatt 1987). While more general and circumventing the need for large waveform databases, parametric models often sound less realistic than CTTS (Schwarz 2004, pp. 15–22).

There have been many analogous applications of CSS-like techniques to sound and music synthesis (Schwarz 2006). As discussed above, some composers have used similar techniques by manually splicing pieces of analog magnetic tape or pasting digital samples. *Granulation* (Roads 2001, pp. 187–193), a synthesis method similar to CSS, is a type of granular synthesis that uses grains derived from a recorded waveform. A close relative of granulation is *brassage* (Wishart 1996, p. 53), which generally focuses on the scrambling of a sound. These techniques have been used to create non-repetitive environmental “soundscapes,” e.g., rain and crickets, from shorter sound examples (Hoskinson 2002). A more complex implementation considers the statistics of the source, which can, for example, synthesize crying babies from a short recording of a single crying baby with little perceived repetition (Dubnov et al. 2002).

In a slightly different application, Nick Collins has devised a program that automatically segments sound files into beats, and then it concatenates these to create different styles of electronic dance music (Collins 2003, 2006). Simon et al. (2005) have performed outstanding work in the synthesis of novel realistic performances by concatenating segments of unrelated performances. They use MIDI and stylistic rules to inform the selections, transformations, and concatenations of sound segments. Finally, Tristan Jehan has used his perceptual sound segmentation scheme (2004) to create “music cross-synthesis,” where segments of one song are concatenated to imitate another (2006).

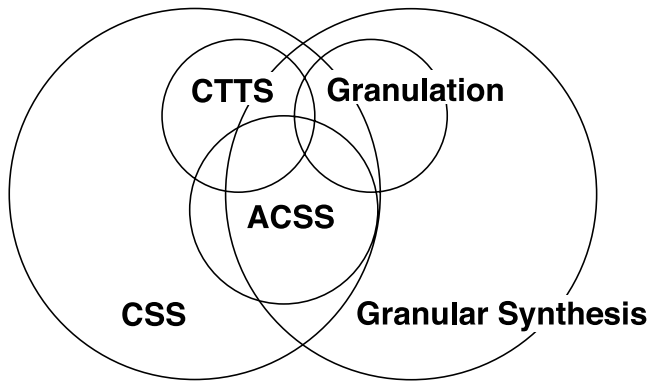
Recent work has investigated and evaluated adaptive concatenative approaches for synthesizing sound and music. Like CTTS, sounds are selected and concatenated according to features (or descriptors) of some “target,” but instead of using text, the target can be a set of rules, a symbolic score, or an-

other sound. All of these methods have been termed more generally “data-driven concatenative sound synthesis” (Schwarz 2004). This article concentrates on the approach where, like “photomosaics” (Silver 2000), a target sound is assembled from other sounds based on a measure of similarity between them. For instance, a speech waveform can be approximated by concatenating units of saxophone in a way that minimizes the mean squared error of the original and synthesis. In this way, the analysis “adapts” to features of the target, much like adaptive signal expansions over redundant bases (Mallat and Zhang 1993). ACSS can be seen as an “adaptive digital audio effect” (Verfaille and Arfib 2001; Verfaille 2003), where features of one sound are used as control parameters for the synthesis of new sounds or the transformation of old ones, whether using concatenation, frequency modulation (Poepel and Dannenberg 2005), or other methods.

It can be argued that ACSS is not really synthesis per se, but instead a type of effect. In one sense, ACSS is granular synthesis or multiple-wavetable synthesis, and in another it is sound scrambling (*brassage*), or a sophisticated version of remixing. The line between synthesis and effect is difficult to draw when the sound segments used are brief. The convention adopted here is that when the result bears little resemblance to the original, then the transformation is more synthesis than effect.

It may be helpful at this point to discuss the similarities and dissimilarities of the synthesis methods already mentioned. Figure 1 is a Venn diagram depicting the relationships between CSS (generalized concatenation of units), ACSS (unit selection, transformation, and concatenation based on features of a target sound), CTTS (concatenation of real speech waveforms to create intelligible speech from text), granular synthesis (creation of sound generally using waveforms of duration 1–100 msec; Roads 2001, p. 86), and granulation (granular synthesis using grains derived from recorded sounds). Obviously, ACSS and CTTS are subsets of CSS, granulation is a subset of granular synthesis, and each one overlaps the others. Whereas granular synthesis focuses on creating textures from varying densities of grains with short durations, CSS generally sequences longer-duration sound units in a monophonic man-

Figure 1. Venn diagram of relationships between five synthesis methods: concatenative sound synthesis (CSS), adaptive concatenative sound synthesis (ACSS), concatenative text-to-speech (CTTS), granular synthesis, and granulation. ACSS can be seen as monophonic granular synthesis when sound segments are 1–100 msec (Roads 2001, p. 86).



ner. CTTS, a specialized subset of CSS, shares elements with all the others: it too can be controlled in some respects by a given target sound through analyses of real speech to demarcate units and ascertain inflection. Moreover, its use of real speech waveforms is not unlike granulation. ACSS can be seen as sequential or monophonic granular synthesis, and as CSS driven by features of a sound; but it can be differentiated from granular synthesis when it uses waveforms longer than 100 msec.

### Implementations of ACSS

At the time the author’s research began, only a few implementations of ACSS had been created: *Musaicing* (Zils and Pachet 2001), *Caterpillar* (Schwarz 2000, 2003, 2004), *MoSievius* (Lazier and Cook 2003), and *Soundmosaic* (Hazel 2003). Since about the year 2000, there has been growing interest in applying concatenative techniques to sound and music synthesis (Schwarz 2006), though not always in an adaptive way (Freeman 2006).

*Musaicing* is proposed as an efficient way to explore and use large sample libraries in music production (Zils and Pachet 2001). Here, a “sample” refers to a sound object that has various high-level descriptors attached, such as instrument, playing technique, tempo, etc. By specifying high-level properties of a desired sequence, appropriate samples are automatically retrieved from a library, transformed, and sequenced, thus saving much time and effort. Using a target sound to provide these

high-level properties, *Musaicing* can produce imitations of it using any collection of samples. The problem is posed as one of costs and constraints. Samples are selected and sequenced according to how well each satisfies local and global constraints while minimizing a global cost. A “*musaic*” is built by iteratively minimizing the cost of satisfying two types of constraints: segment and sequence. Segment constraints are local and include how similar a sample is to a portion of the target. Sequence constraints are global and control the overall combination of samples, for instance continuity among selected samples.

Costs are computed with a distance function of a set of discrete low-level descriptors, which can include mean pitch, loudness, “percussivity,” and spectral characteristics. The global cost is a weighted combination of these costs. Finding the optimal solution, i.e., the one that produces a minimum global cost, requires an exhaustive search and is impractical. Instead, *Musaicing* uses a non-exhaustive method, called adaptive search, to find solutions that are satisfactory for the purposes at hand. Recent work (Aucouturier and Pachet 2006) has extended this approach to real-time interactive ACSS.

Inspired by the prospect of large databases of sound and music, *Caterpillar* (Schwarz 2004) is perhaps the most elaborate implementation of ACSS. Like CTTS, it aims for high-quality synthesis, but of instrument sounds. Unique to *Caterpillar* is the way it defines and uses heterogeneous units. Instead of using a fixed analysis, it attempts to extract meaningful units by dynamic time-warping with a score. Each unit can be further parsed into attack, sustain, and release. Several descriptors are computed for each unit: continuous features (varying over the unit) such as pitch, energy, spectral tilt, spectral centroid, and inharmonicity; discrete features (constant over the unit) such as time location, sound file, and duration; and symbolic features such as phoneme for a voice signal, instrument, notes played, and expressive performance instructions. As in *Musaicing*, units are selected that minimize a global cost while satisfying specified constraints, but another cost is introduced based on the context of a unit and the context of where it is to be placed. For instance, placing a transient-type unit into a

---

steady-state region will have a high cost, even though it might have zero cost for all other constraints. Originally, Caterpillar used a path-search method with  $k$ -nearest neighbors (Schwarz 2000), but the problem was reformulated to use adaptive search (Schwarz 2003).

One complication with these two implementations, Musaicing and Caterpillar, is in determining a set of weights that give “good” results. In both cases, this is done by hand, but a learning algorithm is an attractive alternative (Schwarz 2003). Nonetheless, a user has complete control over how closely and in what ways the output resembles the target.

MoSievius (Lazier and Cook 2003) is a generalized real-time framework for CSS in which a user specifies what discrete features to match and can directly control the synthesis using an input device such as a keyboard. For a faster unit selection process than can be provided by adaptive search, they propose a concept called the “Sound Sieve,” which is a region in a multi-dimensional feature space defined by the user or from the features of a given sound. Only units that exist in this region are considered possible matches, thus reducing the time spent searching.

Soundmosaic (Hazel 2003) takes a much different approach than these other implementations. After segmenting two sounds into homogenous units, comparisons are made between them using the inner product. For each pair having the largest inner product, the two units are switched. No windowing or overlap is used, predictably resulting in choppy and discontinuous output. While the inner product can be optimized in the frequency domain, performing an exhaustive search over all pairs of units makes the algorithm very slow.

An impressive implementation of ACSS that appeared very recently is Scrambled Hackz by Sven König (2006). This software, which will be free and open source, allows one to concatenatively synthesize in real-time, audio input from popular music synchronized to the respective music video frames. A video demonstration by Mr. König succinctly describes and demonstrates his application (König 2006).

Currently, with the exception of Soundmosaic, none of these implementations are available for ex-

ploring ACSS. Though Soundmosaic is freely available, its slow speed and lack of flexibility in selection criteria limits its usefulness for the creative exploration of ACSS. Furthermore, only a few illustrative sound examples created with these implementations can be heard (Hazel 2003; Schwarz 2004; Zils 2006). These reasons motivated the creation of a new implementation that is fast and flexible for creative use, and accessible and available to others interested in exploring ACSS.

### **MATConcat: ACSS Using MATLAB**

MATConcat (Sturm 2004, 2006a) is a free and open-source implementation of ACSS written in MATLAB. To use it, a working installation of MATLAB (preferably version 6.0 or higher) is required. MATLAB was selected because of its cross-platform compatibility, its graphical user interface (GUI) development environment, and its continued use as a tool for teaching media signal processing (Sturm and Gibson 2005).

The MATConcat algorithm, illustrated in Figure 2, is simple and effective. A user-specified window and hop size is used to create homogenous units of audio, of which six discrete descriptors are computed (see Table 1). This creates a six-dimensional feature vector for every unit of audio. This data is stored in matrices, herein loosely referred to as databases, which can be quickly searched by MATLAB. The *corpus* database contains descriptors of units to be selected, as well as pointers to their locations in the original sound files. The *target* database contains descriptors of the units that will control the synthesis. Matches are made between the two databases based on user-selected descriptors and ranges of values, much like the “Sound Sieve” in MoSievius. In more formal terms, similarity between two  $k$ -dimensional feature vectors is judged by a weighted  $l_1$ -norm of their difference. The corpus vector closest to the target vector within a search region is selected. Figure 3 shows an example of the selection process in a two-dimensional feature space.

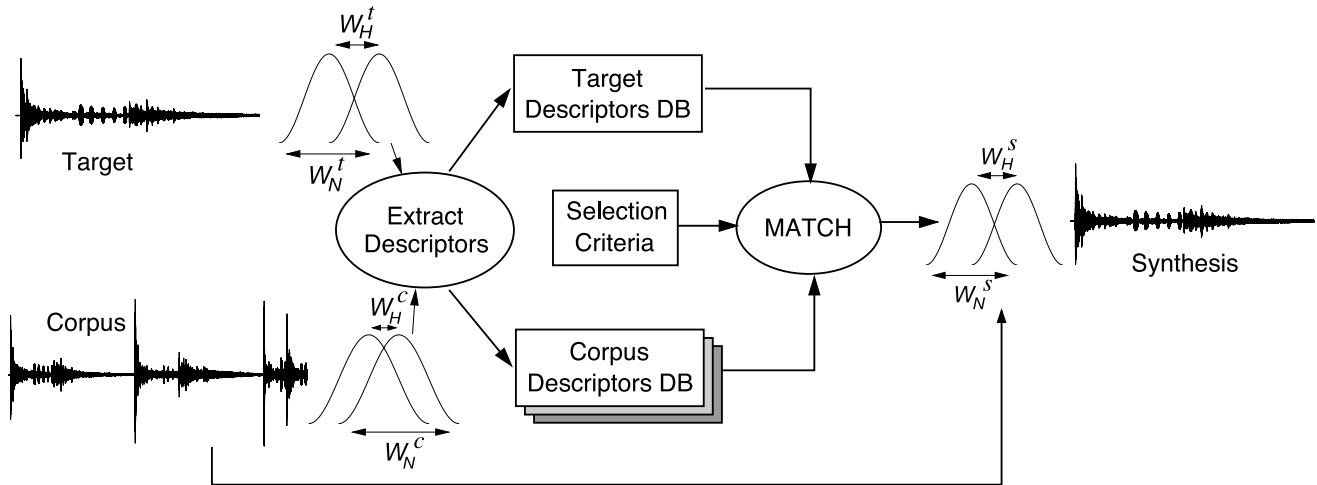
Many descriptors are available to quantitatively describe audio data in both the time and frequency

Figure 2. The MATConcat algorithm. A target sound and collection of corpus sounds are analyzed using a window with a uniform hop size. This creates a database of descriptors for

each unit (window) of sound. In sequential order, each target unit is compared to the corpus database, and a selection is made based on the measure of similarity specified

by the selection criteria. The matching corpus unit is extracted from the relevant sound file, windowed, and added to the synthesis waveform. Note the synthesis can be done using a

window size ( $W_N^S$ ) and hop ( $W_H^S$ ) different from those used for the target and corpus analyses.



**Table 1. Discrete Descriptors Used in MATConcat**

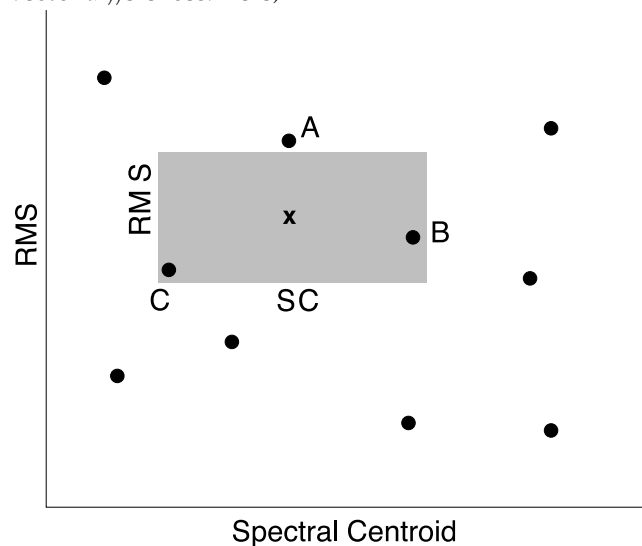
Descriptor	Significance	Formula or Algorithm
Zero Crossings	General noisiness	$ZC = \frac{1}{2} \sum_{n=0}^{N-1}  \text{sgn}(x[n]) - \text{sgn}(x[n-1]) $ , where $\text{sgn}(u) = \begin{cases} -1, u < 0 \\ 1, u \geq 0 \end{cases}$
RMS	Approximate loudness	$RMS = \frac{1}{N} \sqrt{\sum_{n=0}^{N-1} (x[n])^2}$
Spectral Centroid	“Brightness” of sound	$SC = \frac{\sum_{k=1}^{N/2-1} k  X[k] ^2}{\sum_{k=0}^{N/2-1}  X[k] ^2} \frac{F_s}{N}$
Spectral Rolloff	General distribution of energy	$SR = R \frac{F_s}{N}$ , where $R$ is found from: $\sum_{k=0}^R  X[k] ^2 \geq 0.85 \sum_{k=0}^{N/2-1}  X[k] ^2$
Harmonicity	Degree of harmonic relationships among significant partials	Value of second peak of normalized autocorrelation
Pitch	Fundamental frequency	Arfib, Keiler, and Zölzer 2002, pp. 339–341

The descriptors are used to quantitatively describe each unit  $x[n]$ ,  $0 \leq n < N - 1$ , windowed from a real signal sampled at a rate  $F_s$ . Assume  $N$  even.  $X[k]$  is the discrete Fourier transform of  $x[n]$ .

domains (Wold et al. 1996; Casey 2001; Arfib, Keiler, and Zölzer 2002; Tzanetakis 2002, pp. 24–53; Pope, Holm, and Kouznetsov 2004; Schwarz 2004, pp. 85–108). Currently, MATConcat uses the six discrete descriptors shown in Table 1, which are relatively low-level. These descriptors were selected because they are simple, perceptually significant, and simple to implement.

Zero-crossings (the number of times a waveform’s adjacent sample values change sign) is related to the “noisiness” of a signal. Root-mean-square (RMS) is the square root of the sum of the squared sample values, and it provides an approximate idea of loudness. Spectral centroid is the frequency below which half the energy is distributed and is related to the perceived brightness of a sound. Spectral rolloff

Figure 3. An example of searching for vectors (•) similar to a target (x) in a two-dimensional feature space of root-mean square (RMS) and spectral centroid (SC). Similarity is judged by weighted  $l_1$ -norms of vector differences. Here,



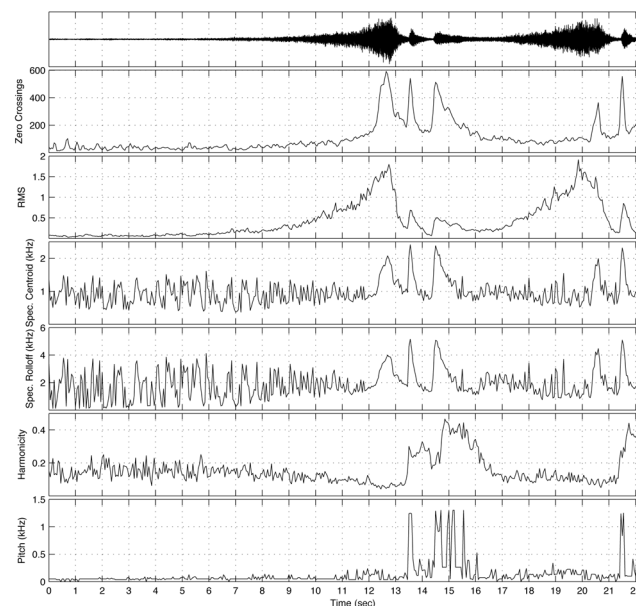
vectors that fall within a region of  $\Delta SC \times \Delta RMS$  centered on the spectral centroid and RMS of the target are possible matches (B and C). Vector B is selected even though A is closer in a Euclidean sense.

is the frequency below which 85 percent (or in some references 95 percent) of the energy is distributed, and provides general information about how the spectrum is distributed. Harmonicity, a measure of how harmonic a waveform is, is found from the value of the second peak of a normalized autocorrelation (Arfib, Keiler, and Zölzer 2002, p. 367). This is a common technique for determining if a segment of speech is voiced, such as a vowel. Finally, the pitch of a unit is determined by finding the first major peak in the spectrum, and fine-tuned using phase information (pp. 339–341).

These six descriptors are shown in Figure 4 for a 1988 recording of the crescendi from the finale of Mahler's second symphony. One channel of the original waveform is shown at top. It can easily be seen that the RMS follows the amplitude trend of the waveform. The spectral centroid and harmonicity reveal the parts of the crescendi that are more pitched than at other times. At these moments there are four trumpets, four trombones, three bassoons, and a contrabassoon playing tutti. Leading up to the climaxes are tremolos on timpani and rolls on a snare drum and tam-tam. The apparent failure of the pitch algorithm is probably due to the percussion; but it also points to the need of a more robust algorithm, perhaps using harmonicity to demarcate regions where pitch is meaningful. It might seem

Figure 4. Distributions of six discrete descriptors of a recording (Mahler 1988) of the crescendi from the finale of Mahler's second

symphony (top). Hann windows of duration 100 msec (4,410 samples) and 50-percent overlap were used.



that spectral centroid and rolloff are correlated, but this is not necessarily so. One can imagine several different spectral distributions that have the same centroid but very different rolloffs.

In MATConcat, any combination of these descriptors can be matched, but as this number grows, the probability of finding satisfactory units obviously becomes smaller unless the corpus grows in size. When a match is found, the corpus unit is extracted from the relevant sound file, windowed, and added to the synthesis in place of the original target unit. The synthesis can be done using several window types, such as Hann and Tukey. This process is repeated for a specified set of target units. Unlike Caterpillar and Mosaicing, MATConcat has no concept of context or concatenation costs; it cares only whether a corpus and target unit have specific descriptors within given ranges of each other.

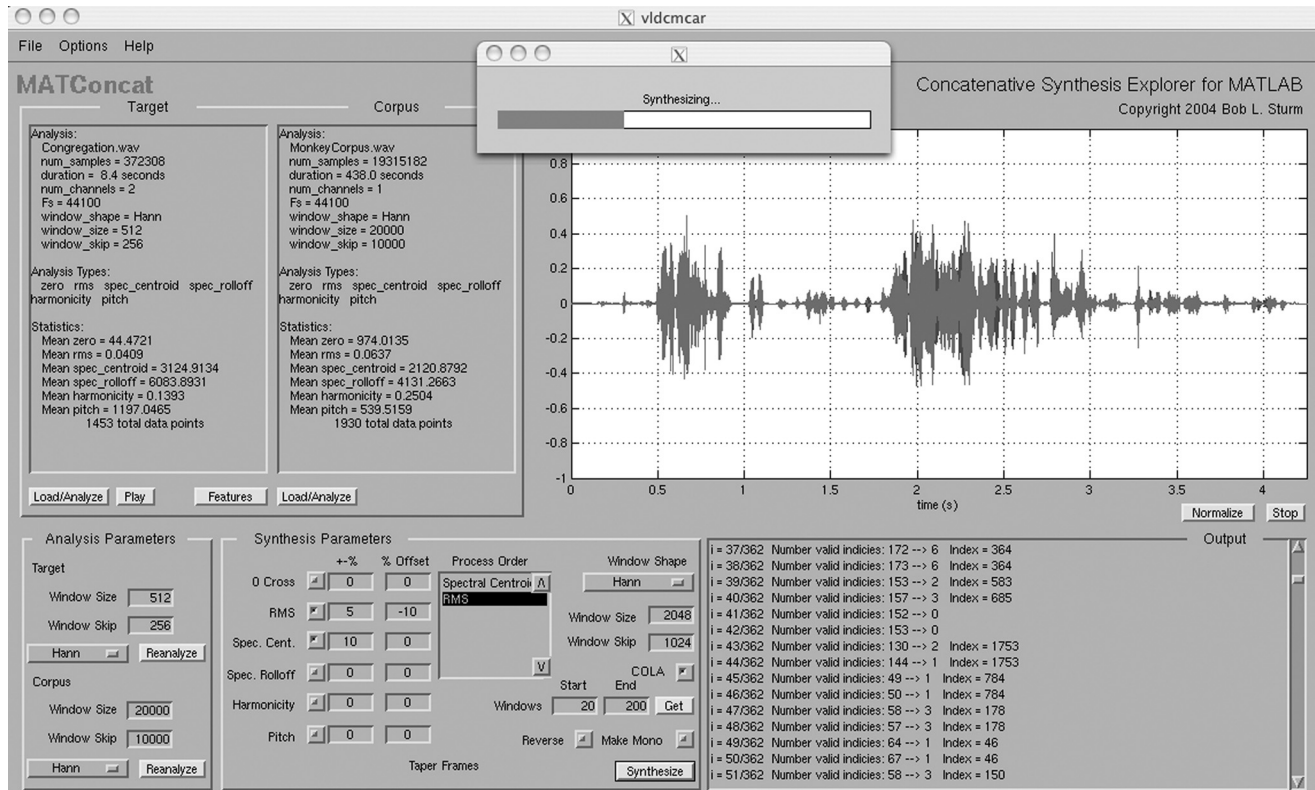
To make it more manageable and accessible, MATConcat is completely controlled using a GUI, a screen shot of which is shown in Figure 5. From the "File" menu, one can open or create a target or corpus database, as well as save analysis databases and results. Basic information about the target and corpus analysis databases is shown in the top-left panes, including total duration of sound material,

Figure 5. Screen shot of the GUI to MATConcat, in the process of synthesizing. The top-left panes show information about the target and corpus databases. The bottom-left pane directs

analyses. The bottom-middle pane provides the selection criteria and parameters for the synthesis. Once a synthesis is finished, the waveform is displayed in the top right, and

matching results are shown in the lower right. Under the "File" menu, one can load, create, and save databases and synthesis results. The "Options" menu provides

directives for unit selection and transformation, such as forcing the RMS of the target onto the synthesis, and extending previous matches if no satisfactory match is found.

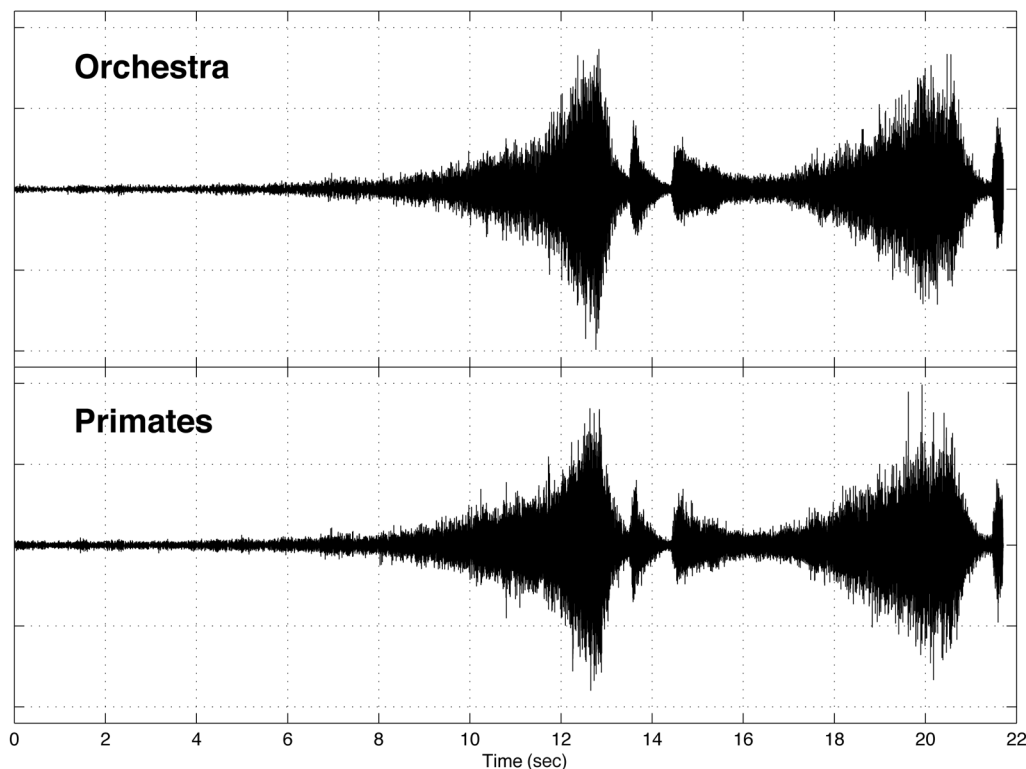


number of units, and basic statistics. Each database can be reanalyzed with settings in the lower-left pane. Synthesis parameters are specified in the lower middle pane. Any set of descriptors can be selected, ranges (weights) and offsets given, and synthesis parameters given. Here, the selection criteria are to first match the spectral centroid within  $\pm 10$  percent of the target value, and then the RMS within  $\pm 5$  percent of 90 percent of the target RMS. Obviously, the smaller the range the larger the weight is given to that feature dimension (see Figure 3). The concept of range is used because it is less abstract than weights. After the process runs, the matching results are shown in the lower-right pane with the synthesis waveform displayed at the top-right. The matching results help guide the selection of good descriptor ranges. As seen in the bottom-right pane of Figure 5, for target unit 40 there are 157 matches that satisfy the spectral centroid condition, and three of these

match the RMS condition. Of these, corpus unit 685 is selected and placed in the synthesis using a window size of 2,048 samples, and an overlap of 1,024 samples (specified in the synthesis parameters pane). Because this corpus is analyzed with a window size of 20,000 samples, only the first 2,048 samples are used. This can of course be changed with the synthesis parameters. For the next target unit, no acceptable corpus unit is found, and thus it is left blank in the synthesis because the option to fill the gap is not selected (hidden from view in the Options menu). There are several options available in MATConcat for different outcomes of the selection process. If no match is found, the program can do nothing, find the next best match, or extend the previous match by adding the next corpus unit; if several matches are found, the program can select the best one, or choose one at random. There are also synthesis options that reverse selected corpus units, convolve



Figure 6: The waveform of Mahler's *crescendi*, mentioned previously, performed by the London Symphony Orchestra (Mahler 1988) is shown above an interpretation by primates (Sturm 2006a, Sound Example A1).



target and corpus units, and force the RMS of target units on corpus units—in effect transforming the corpus unit to match the loudness of the target unit. All of these options can create quite different effects.

Using a plethora of targets and corpora, many examples (Sturm 2006a) have been generated with MATConcat. Several of these will now be presented to demonstrate its performance.

### Examples Generated Using MATConcat

Mahler's percussion *crescendi*, mentioned previously, provide a dynamic set of features (Figure 4). This target has been synthesized with sounds of primates (Sturm 2006a, Sound Examples A1 and A2), a chanting Muslim imam (Sound Example A5), an hour of John Cage's vocal music (Sound Example A6), three hours of music from the *Lawrence Welk Show* (Sound Example A7), and the four string quartets of Arnold Schönberg (Sound Example A8). Each

of these creates entirely different sonic experiences; and depending on the selection and synthesis parameters and options, the impressions of Mahler's *crescendi* can remain or be completely obscured.

For the first example with primates (Sound Example A1), the target is analyzed using a Hann window of duration 46 msec and 50-percent overlap; the corpus is analyzed using a Hann window of duration 372 msec and 93-percent overlap, providing a large number of units to search. The RMS and spectral rolloff descriptors are matched to within  $\pm 5$  percent and  $\pm 10$  percent, respectively. The synthesis is made using Hann windows of the same duration and overlap as the corpus analysis. The primates "ape" the slowly building *crescendi*, creating a sense of increasing hysteria (see Figure 6). At each climax, a dominant gorilla grunts as lesser simians scurry. The primates follow the energy and general spectral trend of the target quite well. By using a smooth and sufficiently large synthesis window, the result is realistic. Though the corpus consists of less

---

than fifty samples of primates, none of them sound repeated—probably due to the redundancy of the corpus created by the small window hop. Quadrupling the synthesis window skip creates crescendi four times as long (Sound Example A2).

These examples demonstrate the applicability of ACSS for generating realistic, non-repetitive and dynamic environments from sound recordings, as done with granulation by Hoskinson (2002) and Dubnov et al. (2002). The difference with ACSS is that field recordings of crickets, birds, or breaking ocean waves, for example, can be choreographed to a target. These sounds can be concatenated to move through different states of activity, for instance agitation and peace, or consonance and dissonance.

Speech provides interesting targets with its fluid rise and fall. Two short speech recordings are synthesized by corpora of primates (Sound Examples B1–B3), J. S. Bach's *Partita* for solo flute (Sound Example B6), alto saxophone (Sound Example C3), and three hours of music from *The Lawrence Welk Show* (Sound Example C4). By careful choice of selection criteria and synthesis parameters, the speech can remain "intelligible." A suitably small range of spectral centroid or rolloff can retain much of the sibilance and breathiness of the original, especially when using corpora with breathy sounds, such as saxophones or flutes.

When using speech as the target and corpus, alien languages appear. This method was used to generate speech sounds for a short animation featuring a gnome. Once suitable selection criteria and synthesis parameters were found, a long target sound of speech was concatenatively synthesized to generate several minutes of material for the gnome speech (Example G1). It is extremely difficult to find the proper selection criteria and synthesis options to produce speech that does not sound scrambled. Without attention to the pitch or spectral centroid, unrealistic speech contours are created; ignoring energy information results in unrealistic emphasis. All of these parameters must be used, but in order to find any matches, the ranges must be relaxed ( $\pm 5$  to 10 percent). In addition, the window size and hop must be chosen such that recognizable words do not appear, the speech does not unnaturally overlap, and the gnome does not sound overactive.

Because of its use of homogenous units, fixed segmentation, and memory-less descriptors, the ability of MATConcat to handle a polyphonic target is unpredictable. The beginning of Arnold Schönberg's fourth string quartet (Schönberg 1939, mm. 1–5) provides an interesting example. The first violin plays the main theme, punctuated by the other players (see Figure 7). The target analysis and synthesis are performed using a Hann window of duration 93 msec and 50-percent overlap. The first eight seconds of this movement are synthesized using alto saxophone with a pitch range of  $\pm 1$  percent of the target (Sound Example D3). In this example, if no matches are found, the previously selected unit is extended until the next successful match.

Figure 7 shows the original waveform and sonogram aligned to the score and the sonogram of the synthesis. Only at times 0.0–0.5 sec and 4.2–5.0 sec does there appear to be any success of the corpus matching the theme in pitch, which occurs precisely when only one instrument is playing. All other moments are marked by attempts to accommodate the transients of the strings playing *marcato*. In the synthesis, the theme is rendered quite discontinuous, but it can be heard with effort. Many of the nuances, like the vibrato, are also missing from the synthesis. Though the theme does not remain intact, ACSS still produces aurally and musically interesting results (Sound Examples D1–D3).

### Comparisons of MATConcat to Other ACSS Implementations

Directly comparing MATConcat with the implementations of ACSS discussed previously is difficult owing to their differences in research goals, not to mention unavailability of code and sound examples for audition. Whereas Musaicing is motivated by finding efficient means for searching and using large sample libraries, Caterpillar is motivated by the promise of high-quality musical synthesis from concatenative techniques. Whereas MoSievius attempts to bring interaction and improvisation to real-time audio collage, Soundmosaic is rough experimental software exploring an interesting idea. All of these implementations, including

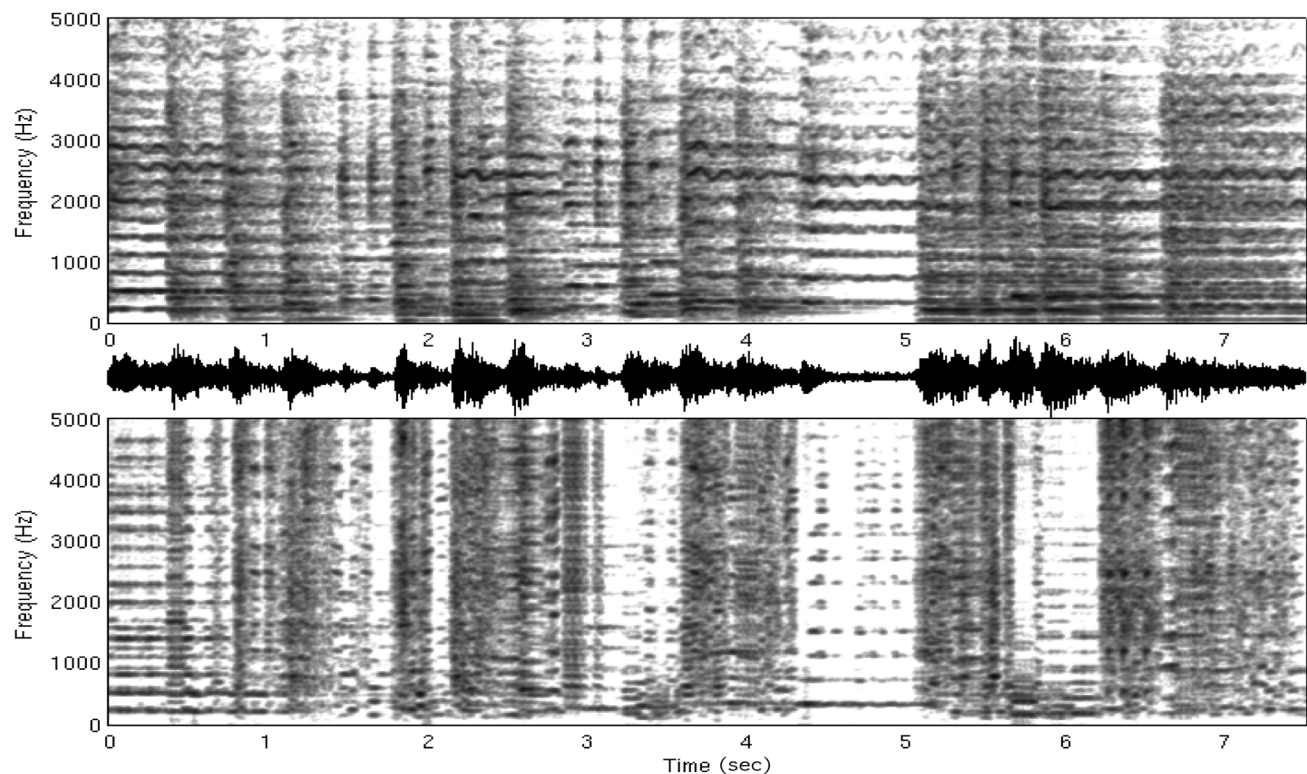
Figure 7. The introduction to Arnold Schönberg's fourth string quartet (Schönberg 1939, mm. 1–5) is aligned (by hand) with the sonogram (top) and waveform (middle) of the recording (Schönberg 1994). Below these is a sonogram

of the result of ACSS (Sturm 2006a, Sound Example D3) matching pitch ( $\pm 1$  percent) using alto saxophone with units extended if no match is found. Hann windows of 93 msec duration and 50-percent overlap were used for both the

analysis and synthesis. For the sonograms, Hann windows of 46 msec were used with 75-percent overlap. ACSS is successful at matching pitch when the first violin is unaccompanied. At other times, it attempts to accommodate

the transients created by the strings playing *marcato*. (String Quartet No. 4 by Arnold Schönberg copyright 1939 [renewed] by G. Schirmer, Inc. [ASCAP]. International copyright secured. All rights reserved. Reprinted by permission.)

**Allegro molto; energico**  $\text{♩} = 152$



**Table 2. Comparison of ACSS Implementations, Inspired by Schwarz (2006)**

<i>Name</i>	<i>Segmentation/ Unit type</i>	<i>Descriptors (Features)</i>	<i>Selection Process</i>	<i>Concatenation Type</i>	<i>Code Language</i>	<i>Speed</i>	<i>Sound Examples</i>	<i>Source Code Available</i>
Musaicing (2001)	Pre-defined/ homogeneous	Low-level discrete	Adaptive search, global and local constraints	?	?	?	Some	
Caterpillar (2000, 2004)	Alignment/ heterogeneous	High- and low-level continuous, discrete, symbolic	Path-search or adaptive search, global and local constraints	Slight overlap with crossfade	MATLAB	?	Some	
Soundmosaic (2003)	Fixed/ homogenous	None	Maximum inner product	Direct substitution	C++	Very slow	Some	✓
MoSievius (2003)	Blind/ heterogeneous	Low-level discrete	Local search	User-defined	C++	Real time		
MATConcat (2004)	Fixed/ homogenous	Low-level discrete	Local search	User-defined, windowed	MATLAB	Fast	Many	✓

Blind segmentation uses audio features for segmentation, whereas alignment segmentation uses a symbolic score to demarcate units. Fixed segmentation cuts the sound into equal sized, or homogenous, units, and does not attempt to relate them.

MATConcat, do however share a common interest in using large databases of sound for creative applications. Table 2 attempts to compile available details of each implementation as accurately as possible.

MATConcat is as much a proof-of-concept and demonstration of ACSS as it is a tool for generating material for electroacoustic music. It creates variations of sounds rather than approximations optimized in some sense. MATConcat is free to use for anyone with access to MATLAB, and many examples of its output are available (Sturm 2006a). Wrapping MATConcat in a GUI facilitates experimentation and data manageability. Performing ACSS from the command line or using a script can quickly become burdensome by having to juggle the parameters and options, and keeping track of results. The simple and informative interface significantly aids in producing results. Furthermore, because it is programmed in a very modular way, MATConcat can be easily extended to handle other discrete descriptors, synthesis parameters, and options.

Even though it is implemented in MATLAB—generally considered to be slow and memory-intensive—it is fast in its analysis and synthesis. Choosing the strategy of matching discrete descriptors within specified ranges, like “Sound Sieve,” significantly decreases the time and overhead required to produce good results, and circumvents the need for linear programming or adaptive searches. It is also much faster than computing and searching correlations. An attempt to use Soundmosaic to synthesize a 4-sec, monophonic target at 44.1 kHz sampling rate, using 0.1-sec units selected from a 37-sec corpus (also monophonic and sampled at 44.1 kHz), took almost 11 hours using default settings (using a 1.25 GHz G4 Apple Powerbook). This limits experimentation to short sounds at low sample rates. Using MATConcat on the same computer, the total analysis time for the same signals using the same window size takes a little over 45 sec. To synthesize the target searching over one descriptor takes about 15 sec. This time is not significantly affected by increasing the number of descriptors to

---

compare. Furthermore, the target and corpus do not need to be analyzed repeatedly; the analyses can be saved for later use. This is essential when working with a large number of sound files. An analysis of about three hours of CD-quality audio using a 46-msec window and 50-percent overlap takes about 8 hours and results in over 380,000 feature vectors. Once saved, this information can be quickly loaded and searched.

An obvious shortcoming of MATConcat is its lack of richness in demarcating and describing units. Creating heterogeneous units using score alignment, such as done in Caterpillar, segmentation from changes in sound texture (Tzanetakis 2002, pp. 67–71), or perceptual models (Jehan 2004) is more meaningful and less arbitrary than the uniform windowed approach used here. A richer set of descriptors, including ones with memory such as differences between units, would certainly aid in interpreting and working with all signals, including polyphonic ones. Furthermore, additional options in the transformation of units would help preserve important aspects of the original, such as string vibrato.

Like the implementations reviewed, MATConcat is a product of research into using this technique for sound synthesis, sparked by the interesting idea of automated collage with large sound databases. Though MATConcat is neither meant to be a general-purpose CSS framework like MoSievius nor an improvement upon the work of others, it is an attempt to make ACSS accessible and applicable to computer music composition. Its application to this end will now be presented.

### **Micromontage Composition Using ACSS**

The applicability of ACSS to computer music composition should be quite clear by now. It presents numerous possibilities for timbrally transforming any sound. A composer can create a target sound, for instance using a variety of vocal sound effects, that provides a stencil on which any sound material may be dabbed. The labor necessary to do this manually with arbitrary precision has been replaced with gathering sounds, exploring parameters and

options, and selecting outcomes. When the sound units extracted are short, the composed results can stylistically be called *micromontage*. It should be emphasized though that micromontage is not ACSS; the former is a style and technique, whereas the latter is an algorithm for sound synthesis and transformation.

In creating micromontage, all of the composers discussed above work with sound material at levels of detail requiring incredible patience and strategy (Roads 2001, pp. 184, 313). Composing with ACSS to achieve similar effects is a much different experience. In contrast, ACSS provides an efficient way to work in the style of micromontage. More time can be spent experimenting, generating, and composing with results than ripping, identifying, segmenting, cataloging, importing, transforming, and finally arranging samples. Moreover, the amount of sound material one can work with is unlimited. Several micromontages created using ACSS and MATConcat are now presented and analyzed.

#### ***Dedication to George Crumb: American Composer***

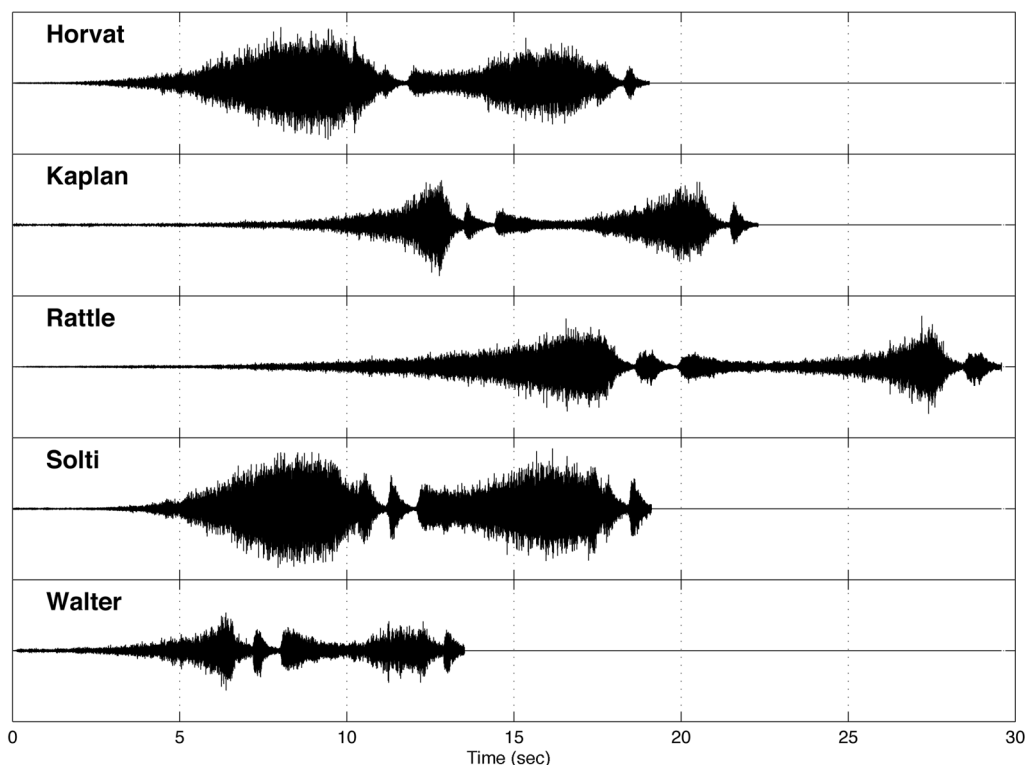
The first test of the applicability of ACSS and MATConcat to composition is the three-movement work *Dedication to George Crumb: American Composer* (Sturm 2006a). A short work by George Crumb is used as the target in all movements. Several hours of Native American music are grouped into three corpora: solo voices, aerophones, and groups. Each movement is assembled from concatenated material from one of these corpora. Once the piece was finished, however, that the lack of an experimental record—the parameters, observations, and results made—could not shed light on the process of composition using ACSS. A new piece was begun and a detailed log of its progress kept.

#### ***Concatenative Variations of a Passage by Mahler***

The composition *Concatenative Variations of a Passage by Mahler* (CVM; Sturm 2006a) systematically explores the idea of timbrally transforming a passage in an electroacoustic context. CVM uses as

Figure 8. Waveforms of interpretations of the crescendi from Mahler's second symphony by five different conductors (Mahler 1987, 1988, 1990,

1991, 1995). Material from these is used as targets or corpora for the composition Concatenative Variations of a Passage by Mahler (Sturm 2006a).



its subject five interpretations (Mahler 1987, 1988, 1990, 1991, 1995) of Mahler's dramatic percussion crescendi mentioned previously. As can be seen from the waveforms in Figure 8, differences among them are quite clear in duration, intensity, and shape. With these "passages" as targets or corpora, MATConcat is used to explore possibilities and generate fine-tuned results. These are recorded, categorized, and arranged and transformed within a multitrack environment to create each variation. CVM currently consists of eleven variations, with several others planned.

Table 3 shows details of each variation, including targets, corpora, and analysis and synthesis settings. For example, the variation "Gates I" uses the interpretation by Simon Rattle (Mahler 1987) as the target and a corpus containing 406 sec of a squeaky gate (recorded by the author at the Rock of Cashel, Ireland) and trumpet (played by the author). The target and corpus are analyzed with 23- and 93-msec windows respectively, with 50-percent overlap in

both cases. The material used in the variation is synthesized with Hann windows of 93- and 227-msec durations, both having skips of 46.5 msec. In one case, the descriptors matched are RMS with a range of  $\pm 2$  percent and spectral centroid with a range of  $\pm 1$  percent. The only option specified is to extend the previously selected unit if no match is found.

The one-minute variation "Boils and Bells" explores very short synthesis windows and uses over 2,800 segments of 46-msec duration from a corpus consisting of more than an hour of sound effects. Done manually, this would obviously require hundreds of hours of work and an uncanny ability to organize. Apparently, the sound effects most similar to Mahler's crescendi are boiling water, bells, a toy cow, jackhammers, and at the height of the first crescendo, a man falling off a ladder.

Many trials are run for each variation to explore the potential of the target, corpus, selection criteria, synthesis parameters, and the available options.

**Table 3. Settings and Parameters for each Variation of *Concatenative Variations of a Passage by Mahler* (Sturm 2006a), in no Particular Order**

Variation	Target and Duration (sec)	Corpora and Duration (sec)	Analysis Window Size/Skip (msec)		Synthesis Window Type Size/Skip (msec)	Selection Criteria ( $\pm\%$ )	Settings
			Target	Corpus			
<i>Passage I</i>	Speech 8	Horvat, Kaplan, Rattle, Solti, Walter 104	3/1.5	46/23	Hann 3628/23	{SC(5%), SR(5%)}, {SR(50%), P(80%)}	RM, EM, FRMS
<i>Passage II</i>	Kaplan 23	Horvat (mm. 193–206) 36	11/5.5	11/5.5	Hann 113/23	{RMS(5%), SC(5%)}	EM
<i>Gates I</i>	Rattle 29	Squeaky gate 192, trumpet 214	23/11.5	93/46.5	Hann 93/46.5, 227/46.5	{RMS(2%), SC(1%)}, {RMS(2%), SR(1%)}	EM
<i>Gates II</i>	Rattle 29 (reversed)	Squeaky gate, trumpet (3 pitch shifted versions) 1621	23/11.5	93/46.5	Hann 93/46.5, 227/46.5	{RMS(1%), SC(1%)}	EM
<i>Creatures</i>	Kaplan 36 (modified)	Various animals 3607, primates 438, birds 3604	46/23	500/23, 372/23, 113/57	Hann 500/23, 372/23	{RMS(5%), SR(10%)}	–
<i>Boils and Bells</i>	Walter 13	Sound effects 4080	12/6	46/23	Hann 46/23	{RMS(1%), SC(1%)}, {RMS(0.1%), SC(1%)}	EM
<i>Saxubus</i>	Solti 19	Solo alto saxophone (Braxton 2000) 1191	46/23	272/136	Tukey 25% 408/272	{SC(0.05%)}	RM
<i>Lix Tetrax</i>	Solti 19 (reversed)	“Partita” for flute by J.S. Bach (Rampal 1992) 658	123/66.5	93/46.5	Tukey 25% 363/227	{RMS(0.5%)}	–
<i>Limbo</i>	Kaplan 23; Walter 14	The Lawrence Welk Show 8855	23/11.5	46/23, 227/113	Tukey 25% 680/23, 680/113	{SC(0.1%), SR(0.1%)}, {RMS(25%)}, {SC(1%) SR(1%)}	EM, FRMS
<i>A cappella</i>	Kaplan 22 (modified)	Solo pop vocal samples 1913	46/23	113/56.5	Hann 113/56.5, 453/56.5	{RMS(3%), SR(3%)}, {SC(1%) P(1%)}	EM, RM
<i>Highway to Heaven, Stairway to Hell</i>	Solti 20	AC/DC “Highway to Hell” 209, Led Zeppelin “Stairway to Heaven” 483	113/56.5	136/68	Hann 136/68	{RMS(0.1%) $\pm$ 50%, +200%}	EM

Key: RMS = root-mean-square, SC = spectral centroid, SR = spectral rolloff, P = pitch, RM = random match, EM = extend match, FRMS = force RMS

Even subtly different selection criteria can produce unique and interesting results that provide fruitful avenues for exploration. The variations “Gates I” and “Gates II” use the same corpus and interpretation by Simon Rattle, but for the latter, the target is

Mahler’s passage reversed. Both variations match the same features, but with slightly altered ranges. It is surprising how different in character they are, considering they come from the same material.

In rare cases, a musically satisfying outcome from

**Table 4. Details of Each Trial Run to Generate Material for *Saxubus*.**

Trial	Analysis Size/Skip (ms)		Synthesis Window Type Size/Skip (msec)	Selection Criteria ( $\pm\%$ )	Options	Comments
	Target	Corpus				
1	93/46.5	93/46.5	Hann 93/46.5	{SC(1)}	–	Too active
2	93/46.5	93/46.5	Hann 93/46.5	{SC(1)}	RM	Too active
3	93/46.5	93/46.5	Hann 93/46.5	{SC(1)}	RM	Too active; not much different from 2
4	93/46.5	93/46.5	Hann 93/46.5	{RMS(1), SC(1)}	–	Less fluid with many gaps; crescendi obvious
5	93/46.5	93/46.5	Hann 93/46.5	{RMS(20), SC(10)}	–	Interesting moments at height of crescendi
6	93/46.5	93/46.5	Hann 93/46.5	{RMS(20), SR(10)}	–	Similar to 5; in general crescendi too obvious
7	93/46.5	93/46.5	Hann 363/46.5	{SC(1)}	–	Interesting; synthesis windows too long
8	93/46.5	272/136	Hann 272/136	{SC(1)}	RM	Nice feel; crescendi not obvious
9	93/46.5	272/136	Hann 272/136	{SR(1)}	RM	Different results from using SC in 8
10	272/136	272/136	Hann 272/136	{SR(1)}	RM	Slightly different from 9
11	272/136	272/136	Hann 272/136	{SC(1)}	RM	Slightly different from 7
12	12/6	272/136	Tukey 25% 272/272	{SC(0.5)}	RM	Tukey window has nice effect; too few matches
13	12/6	272/136	Tukey 25% 408/272	{SC(0.5)}	RM	Longer window, same overlap as 12; nice effects
14	93/46.5	272/136	Tukey 25% 408/272	{SC(0.05)}	RM	More matches with tighter descriptor range and longer target analysis; result getting closer to something musical
15	93/46.5	272/136	Tukey 25% 408/272	{SC(0.01)}	RM	. . . closer . . .
16	46/23	272/136	Tukey 25% 408/272	{SC(0.05)}	RM	This is it!
17	46/23	272/136	Tukey 25% 408/272	{SC(0.05)}	FRMS, RM	Too much like the crescendi

Key: RMS = root-mean-square, SC = spectral centroid, SR = spectral rolloff, RM = random match, FRMS = force RMS

ACSS is immediate, in the sense that the output needs little to no editing. This happened for “*Saxubus*,” a “recomposition” of the interpretation by Sir Georg Solti (Mahler 1991) with a 20-minute corpus of Anthony Braxton playing alto saxophone

(Braxton 2000). Seventeen trials were run to hone in on satisfactory results. Details of each trial are shown in Table 4. The first six trials (Sound Examples E1–E6) give frenetic results, to which the gaps in the fourth trial (Sound Example E4)—cre-



---

ated when no match is found—provide effective contrasts. By extending the synthesis window size but not the skip (Sound Example E7), the results are more fluid, but the sound is too dense. Trials 8–11 (Sound Examples E8–E11) test different analysis window sizes and selection criteria. Trial 12 (Sound Example E12) uses a very short analysis window and is the first to use a long Tukey window for the synthesis. Through a fine-tuning of these settings in trials 14–17 (Sound Examples E14–E17), the result in trial 16 immediately stands out as unique. This result is cropped and arranged with a few short reprises to form “Saxubus.”

From the two channels of the target, an amusing hocketed saxophone duet results. The crescendi transform into gradually increasing activity with occasional contrasting pauses created when no suitable matches are found. The only descriptor matched in trial 16 is the spectral centroid with a very small range of  $\pm 0.05$  percent. Synthesis using fairly long Tukey windows with 25-percent cosine lobes creates shorter attacks and decays as opposed to the gradual fades of Hann windows. The random-match setting specifies that when more than one match is found, the selection from these is random. Together, these settings ensure a diversity of selected units from the corpus. Surprisingly, though the target and corpus are hardly tonal or rhythmic, the selection criterion and synthesis parameters generate a regularly pulsing pedal point.

More often, however, additional work is required to fashion a satisfying musical experience from the output of MATConcat. In the output of the only trial (Sound Example F8) used in “Lix Tetrax,” a flute playing a long and piercing high A is heard. This requires some editing to create a much more pleasing experience. “Lix Tetrax” uses a time-reversed interpretation (Mahler 1991) as the target, and the *Partita* for solo flute by J. S. Bach (Rampal 1992) as the corpus. RMS is matched with a small range of  $\pm 0.5$  percent, and a long-duration Tukey window is used in the synthesis with no extension of units to fill null matches.

Though the same target as “Saxubus” is used, the results are superlatively different in both shape and feel. Mahler’s crescendi are gone, and no portion of Bach can be recognized save for a few mordents. Mr.

Rampal’s performance has been disassembled and reassembled in a way that only his essence remains. The result remains fluid instead of jagged owing to the window shape and size, and it gives the impression of an actual performance, though each corpus unit lasts only 363 msec. Indeed, if this process were synchronized with the score of the *Partita*, then a score of “Lix Tetrax” could be generated in parallel for real performers. The only problem is in concatenating elements of a score as easily as units of sound.

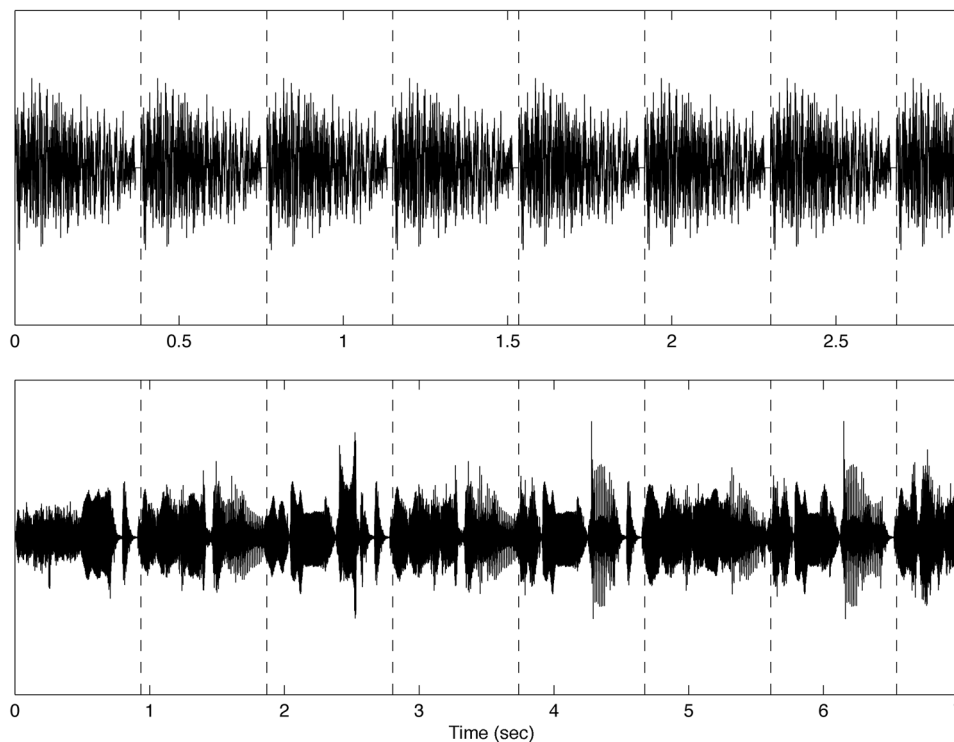
To create forms that are more complex than the crescendo and decrescendo, Mahler’s passage is rearranged to form new targets. The most drastic rearrangements occur in the targets for “A cappella.” One target uses exact repetitions of 400 msec taken from the brass notes of Gilbert Kaplan’s interpretation (Mahler 1988). The first few seconds of this target aligned with the synthesis result are shown in Figure 9. The output of ACSS creates an exciting feeling of rhythm instead of direct repetition. At times it sounds quite realistic, as if performed by vocalists with an uncanny sense of timing. Though the target consists of exact repetitions, it would take 9,200 repetitions of this material before the features began repeating, because the analysis hop is only 23 msec.

The only variations that use the interpretations of Mahler’s passage in the corpus are “Passage I” and “Passage II.” The former uses a target completely unrelated: a man saying “Neither this recording nor any part thereof may be reproduced or used for any purpose without prior written authority from. . . .” The target analysis uses extremely short windows (11 msec), which produces over 5,500 feature vectors. The synthesis uses very long Hann windows (3.6 sec) and a small skip of 23 msec. These settings expand the 8-sec target to over 138 sec in the synthesis. The result is an incredibly dense ebb and flow of drum rolls and horns, sounding like Wagner carried on the winds and waves of a hurricane, as one concertgoer said.

## Discussion

Each variation in CVM serves as a study of some aspect of ACSS, whether it is selection criteria, anal-

Figure 9. Waveform of the first seven seconds of “A cappella” (bottom) and the target used to generate it (top). Dashed lines denote regions of exact repetition in the target. Because the synthesis window hop is 2.45 times the target analysis hop, the synthesis is 2.45 times longer than the target.



ysis and synthesis window and hop sizes, material in the target and corpus, or other options such as extending units when suitable matches are not found. Mahler’s passage is quite appropriate for this implementation of ACSS: It is simple, monophonic, transparent in shape, and has dynamic features. It provides a fertile ground for planting samples. Without the aid of ACSS, it would have taken years to compose CVM from the dozens of hours of material used. Thousands of decisions and hours of work have been assigned to the computer, which is by design more adept than a human at the mechanical operation of numerical comparisons and shuttling of data. That leaves more time to dream of possibilities for future variations.

The incredible amount of labor done by composers John Cage, Iannis Xenakis, James Tenney, Horacio Vaggione, Curtis Roads, Noah Creshevsky, and John Oswald, carefully combining by hand short samples of sound, cannot be reproduced so easily. Noah Creshevsky has remarked that performing this labor has its benefits: Through becom-

ing so familiar with sound material, interesting routes for exploration are made manifest (2005). However, for most variations in CVM, generating material using ACSS is just one step of the compositional process. Prior to this, several trials must be run to assess the fertility of all available options; it is not just a matter of pressing buttons and becoming fortunate. After an intuition of the algorithm is developed, ACSS becomes a rich tool for quickly creating imitations and variations of any given sound for composition.

Because of its use of recorded sound, CSS in general poses interesting legal questions. Of the variations in CVM, only a few contain material that is not copyright-protected. This raises important legal questions, especially when the material is as recognizable as in “Limbo” and “Highway to Heaven, Stairway to Hell.” Though J. S. Bach’s *Partita* for flute is not recognizable in “Lix Tetrax,” or Anthony Braxton’s album *For Alto* in “Saxubus,” the recorded performers are still Jean-Pierre Rampal and Anthony Braxton; and the rights to reproduce any

---

portion of the sound recordings are owned exclusively by them or other entities. An examination of the legal issues of CSS in general is the topic of another article (Sturm 2006b).

## Conclusion

In his “Viewpoints on the History of Digital Synthesis” (1991), Julius O. Smith, III, discusses the shift in synthesis research from abstract mathematical concepts that predominated the early years of computer music to more physically informed and natural models of sound generation. He argues that this has occurred in part because the potential for and ease in crafting interesting sounds increases when parameters for synthesis are intuitive and natural. He writes:

The most straightforward way to obtain interesting sounds is to draw on past instrument technology or natural sounds. Both spectral-modeling and physical-modeling synthesis techniques can model such sounds. In both cases, the model is determined by an analysis procedure that computes optimal model parameters to approximate a particular input sound. The musician manipulates the parameters to create musical variations. (p. 11)

These observations are equally applicable to ACSS, where features of an input sound are approximated by features of other sounds to create interesting variations. In a sense, it can be seen as an extended form of query-by-example (Wold et al. 1996; Tzanetakis, Ermolinskyi, and Cook 2002) applied to music composition. Instead of retrieving a similar piece of audio, it assembles and transforms many pieces of audio into the sound desired—in a sense creating an aural “caricature” of the query (Tzanetakis, Ermolinskyi, and Cook 2002). By interfacing this algorithm with large and efficient databases of audio information (Pope, Holm, and Kouznetsov 2004)—for instance all sound recordings made to date—and using better descriptors and more informed methods of concatenation, a potentially realistic and flexible sound synthesis engine is possible (Schwarz 2004).

Supported by results from MATConcat and other implementations presented here, ACSS provides an efficient way to transform the “brittle, frozen music” of samplers (Smith 1991, p. 8) into effective and expressive music. Indeed, it is quite intuitive and natural for a composer to ask, “Create a sound that goes ‘WEEeewooOW-POP’ but played by a violin and bongo.” With ACSS, this is completely possible: one can synthesize and compose by imitation instead of having to program physically unintuitive and abstract algorithms. Though many sample-level operations have been relegated to the computer, a composer still has the task of directing the algorithm and selecting and arranging the results in meaningful ways.

For any synthesis or transformation method, of course, the proof of the method is in the hearing. Curtis Roads (2001) writes: “Scientific tests help us estimate the potential of a technique of synthesis or sound transformation. They may even suggest how to compose with it, but the ultimate test is artistic. The aesthetic proof of any signal processing technique is its use in a successful composition” (p. 301). Through the sound examples and compositions presented above, it has been demonstrated that even the relatively simple implementation of ACSS in MATConcat creates effective and intriguing sound and music. It presents a wide range of compositional possibilities using thousands of different sounds and thousands of transformations. ACSS serves well as a massive sample mill, grinding sound into minuscule pieces for reconstitution into novel expressive forms.

## Acknowledgments

Thanks to Noah Creshevsky, John Oswald, Diemo Schwarz, Stephen Pope, and my advisor Curtis Roads for their enthusiasm for my work. Thanks to my wonderful wife Carla, who showed as much excitement as I in hearing Mahler interpreted by primates. And thanks to the Editor and anonymous reviewers for many helpful suggestions. This research is supported in part by NSF IGERT in Interactive Digital Multimedia Grant No. DGE-0221713.

---

## References

- Arfib, D., F. Keiler, and U. Zölzer. 2002. "Time-Frequency Processing." In U. Zölzer, ed. *DAFX—Digital Audio Effects*. West Sussex, UK: Wiley, pp. 237–298.
- Aucouturier, J.-J., and F. Pachet. 2006. "Jamming with Plunderphonics: Interactive Concatenative Synthesis of Music." *Journal of New Music Research* 35(1): 35–50.
- Braxton, A. 2000. *For Alto*. Audio compact disc. Chicago: Delmark DE-420. (Originally recorded in 1969.)
- Casey, M. 2001. "MPEG-7 Sound-Recognition Tools." *IEEE Transactions on Circuits and Systems for Video Technology* 11(6):737–747.
- Collins, N. 2003. "Recursive Audio Cutting." *Leonardo Music Journal* 13:23–29.
- Collins, N. 2006. "BBCut2: Integrating Beat Tracking and On-the-fly Event Analysis." *Journal of New Music Research* 35(1): 63–70.
- Creshevsky N. 1995. "Borrowed Time." Recorded on *Auxesis: Works by Charles Amirkhanian and Noah Creshevsky*. Audio compact disc. Centaur Records CRC 2194.
- Creshevsky, N. 2001. "On Borrowed Time." *Contemporary Music Review* 30(4):91–98.
- Creshevsky, N. 2003. *Hyperrealism: Electroacoustic Music by Noah Creshevsky*. Audio compact disc. Mutable Music MU512.
- Creshevsky, N. 2005. Personal communication, 21 March.
- Dubnov, S., et al. 2002. "Synthesis of Audio Sound Textures by Learning and Resampling of Wavelet Trees." *IEEE Computer Graphics and Applications* 22(4):38–48.
- Freeman, J. 2006. "Audio Signatures of iTunes Libraries." *Journal of New Music Research* 35(1): 51–61.
- Hazel, S. 2003. Soundmosaic Web site, [www.thalassocracy.org/Soundmosaic/](http://www.thalassocracy.org/Soundmosaic/) (accessed March 21, 2006).
- Holm-Hudson, K. 1997. "Quotation and Context: Sampling and John Oswald's Plunderphonics." *Leonardo Music Journal* 7:17–25.
- Hoskinson, R. 2002. *Manipulation and Resynthesis of Environmental Sounds with Natural Wavelet Grains*. Master's thesis, University of British Columbia.
- Hunt, A. J., and A. W. Black. 1996. "Unit Selection in a Concatenative Speech Synthesis System Using a Large Speech Database." *Proceedings of the 1996 IEEE International Conference On Acoustics, Speech, and Signal Processing*. New York: Institute of Electrical and Electronics Engineers, pp. 373–376.
- Jehan, T. 2004. "Event-Synchronous Music Analysis/Synthesis." *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx-04)*. Naples: Federico II University of Naples.
- Jehan, T. 2006. Music Cross-Synthesis Examples, [web.media.mit.edu/~tristan/Blog/Cross\\_Synthesis\\_v1.html/](http://web.media.mit.edu/~tristan/Blog/Cross_Synthesis_v1.html/) (accessed March 21, 2006).
- Klatt, D. H. 1987. "Review of Text-to-Speech Conversion for English." *Journal of the Acoustical Society of America* 82(3):737–793.
- König, S. 2006. sCrAmBled?HaCkZ software demonstration, [www.popmodernism.org/scrambledhack/](http://www.popmodernism.org/scrambledhack/) (accessed September 25, 2006).
- Kostelanetz, R., ed. 1970. *John Cage*. New York: Praeger.
- Lazier, A., and P. Cook. 2003. "MoSievius: Feature Driven Interactive Audio Mosaicing." *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx-03)*. London: Queen Mary University of London, pp. 1–6.
- Mahler, G. 1987. *Symphony No. 2*, conducted by S. Rattle. Audio compact disc. EMI Classics 47962.
- Mahler, G. 1988. *Symphony No. 2*, conducted by G. Kaplan. Audio compact disc. MCA Classics MCAD 2-11011.
- Mahler, G. 1990. *Symphony No. 2*, conducted by M. Horvat. Audio compact disc. ZYX Music GMBH.
- Mahler, G. 1991. *Symphony No. 2*, conducted by G. Solti. Audio compact disc. London Records 30804.
- Mahler, G. 1995. *Symphony No. 2*, conducted by B. Walter. Audio compact disc. Sony Classical 64447.
- Mallat, S., and Z. Zhang. 1993. "Matching Pursuit with Time-Frequency Dictionaries." *IEEE Transactions on Signal Processing* 41(12):3397–3414.
- Oswald, J. 1993. *Plexure*. Audio compact disc. Avant 016.
- Oswald, J. 2001. *69plunderphonics96*. Audio compact discs. Seeland Records 515.
- Oswald, J. 2006. Plunderphonics Web site, [www.plunderphonics.com](http://www.plunderphonics.com) (accessed March 21, 2006).
- Parmegiani, B. 2002. *La mémoire des sons*. Audio compact disc. Institut National Audiovisuel, Groupe de Recherches Musicales 2019.
- Poepel, C., and R. Dannenberg. 2005. "Audio Signal Driven Sound Synthesis." *Proceedings of the 2005 International Computer Music Conference*. San Francisco, California: International Computer Music Association, pp. 391–394.
- Pope, S. T., F. Holm, and A. Kouznetsov. 2004. "Feature Extraction and Database Design for Music Software." *Proceedings of the 2004 International Computer Music Conference*. San Francisco, California: International Computer Music Association, pp. 596–603.
- Rampal, J. P. 1992. *Le Flûtiste du Siècle*. Audio compact disc. Erato Classics 2292-45830-2.
- Roads, C. 2001. *Microsound*. Cambridge, Massachusetts: MIT Press.

- Roads, C. 2004. *Point Line Cloud*. Audio compact disc. Asphodel ASP 3000.
- Roads, C. 2005. Personal communication, 21 March.
- Schönberg, A. 1939. *Fourth String Quartet, Op. 37*. New York: Schirmer.
- Schönberg, A. 1994. *Arnold Schönberg 2: Streichquartette I–IV*, performed by the Arditti String Quartet. Audio compact disc. Auvidus MO 782024.
- Schwarz, D. 2000. "A System for Data-Driven Concatenative Sound Synthesis." *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx-00)*. Verona, Italy: University of Verona, pp. 97–102.
- Schwarz, D. 2003. "The Caterpillar System for Data-Driven Concatenative sSound Synthesis." *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx-03)*. London: Queen Mary University of London.
- Schwarz, D. 2004. *Data-Driven Concatenative Sound Synthesis*. Ph.D. thesis, Académie de Paris, Université Paris 6. Available online at [recherche.ircam.fr/equipements/analyse-synthese/schwarz/](http://recherche.ircam.fr/equipements/analyse-synthese/schwarz/) (accessed March 21, 2006).
- Schwarz, D. 2006. "Concatenative Sound Synthesis: The Early Years." *Journal of New Music Research* 35(1):3–22.
- Silver, R. 2000. "Digital Composition of a Mosaic Image." United States Patent #6,137,498.
- Simon, I., et al. 2005. "Audio Analogies: Creating New Music from an Existing Performance by Concatenative Synthesis." *Proceedings of the 2005 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 65–72.
- Smith, J. O. 1991. "Viewpoints on the History of Digital Synthesis." *Proceedings of the 1991 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 1–10. Available online at [ccrma-www.stanford.edu/~jos/kna/kna.pdf/](http://ccrma-www.stanford.edu/~jos/kna/kna.pdf/) (accessed March 21, 2006).
- Sturm, B. L. 2004. "MATConcat: Concatenative Sound Synthesis Using MATLAB." *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx-04)*. Naples: Federico II University of Naples, pp. 323–326.
- Sturm, B. L. 2006a. MATConcat software Web site, [www.mat.ucsb.edu/~b.sturm/research.html/](http://www.mat.ucsb.edu/~b.sturm/research.html/) (accessed September 25, 2006). Audio examples and compositions discussed in this article are available at [www.mat.ucsb.edu/~b.sturm/CMJ2006/MATConcat.html/](http://www.mat.ucsb.edu/~b.sturm/CMJ2006/MATConcat.html/) (accessed September 25, 2006).
- Sturm, B. L. 2006b. "Concatenative Sound Synthesis and Intellectual Property: An Analysis of the Legal Issues Surrounding the Synthesis of Novel Sounds from Copyright-Protected Work." *Journal of New Music Research* 35(1): 23–33.
- Sturm, B. L., and J. D. Gibson. 2005. "Signals and Systems Using MATLAB: An Integrated Suite of Applications for Exploring and Teaching Media Signal Processing." In *Proceedings of the 2005 IEEE Frontiers in Education Conference*. Indianapolis, Indiana: Institute of Electrical and Electronics Engineers, pp. 456–459.
- Tenney, J. 1992. *James Tenney: Selected Works 1961–1969*. Audio compact disc. Artifact Recordings 1007.
- Tzanetakis, G. 2002. *Manipulation, Analysis, and Retrieval Systems for Audio Signals*. Ph.D. thesis, Princeton University.
- Tzanetakis, G., A. Ermolinskyi, and P. Cook. 2002. "Beyond the Query-By-Example Paradigm: New Query Interfaces for Music Information Retrieval." *Proceedings of the 2002 International Computer Music Conference*. San Francisco: International Computer Music Association, pp. 177–183.
- Vaggione, H. 1995. *Chrysopée Electronique–Bourges*. Audio compact disc. Mnémosyne Music Media LDC 2781102.
- Vaggione, H. 2005. Personal communication, 21 March.
- Verfaillie, V. 2003. "Effets Audionumériques Adaptatifs: Théorie, Mise en Œuvre et Usage en Création Musicale Numérique." Ph.D. thesis, Université Aix-Marseille II.
- Verfaillie, V., and D. Arfib. 2001. "A-DAFx: Adaptive Digital Audio Effects." *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx-01)*. Limerick: University of Limerick, pp. 10–14.
- Wishart, T. 1996. *On Sonic Art*. Amsterdam: Harwood.
- Wold, E., T. Blum, D. Keislar, and J. Wheaton. 1996. "Content-based classification, search and retrieval of audio." *IEEE Multimedia* 3(2):27–36.
- Xenakis, I. 1992. *Formalized Music: Thought and Mathematics in Music*. Stuyvesant, New York: Pendragon Press.
- Zils, A. 2006. Musical Mosaicing Web site, [www.csl.sony.fr/~aymeric/](http://www.csl.sony.fr/~aymeric/) (accessed March 21, 2006).
- Zils, A., and F. Pachet. 2001. "Musical Mosaicing." *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx-01)*. Limerick: University of Limerick, pp. 39–42.