

AN INTELLIGENT MULTI-TRACK AUDIO EDITOR¹

Roger B. Dannenberg
Carnegie Mellon University
School of Computer Science

ABSTRACT

Audio editing software allows multi-track recordings to be manipulated by moving notes, correcting pitch, and making other fine adjustments, but this is a tedious process. An “intelligent audio editor” uses a machine-readable score as a specification for the desired performance and automatically makes adjustments to note pitch, timing, and dynamic level.

1. INTRODUCTION

Multi-track recording in the studio is a standard production method for a variety of musical styles, including classical, experimental, and popular forms. Multi-track techniques are used in many recording situations, including ensembles that record all tracks at once, ensembles that record tracks in several passes, and even individuals that play different parts in succession. One of the advantages of multi-track recording is the control it gives *after* the music is recorded. Balance, equalization, reverberation, and many other parameters can be altered without the time and expense of re-recording.

Originally, analog recording allowed individual musicians to correct mistakes quickly by “punching in,” *i.e.*, replacing as little as one note of one track with new material. Digital recording has ushered in many more possibilities. Now, individual notes or whole phrases can be shifted in time, stretched, transposed to improve intonation, or even replaced, perhaps using material from the same performance.

As with many technologies, new capabilities work quickly to establish a new set of expectations. Now, virtually every music recording is carefully edited and refined using digital techniques. Ironically, the efficiency and effectiveness of digital editors often results in much *more* time spent in production than ever before.

Meanwhile, there have been many advances in audio and music analysis, machine listening, audio-to-score alignment, and music understanding. Tasks such as finding notes, aligning them in time, correcting intonation problems, and balancing dynamic levels can be performed automatically, at least in principle, and this is the objective of the present research. In the production version of this work, one starts

with audio recordings and a machine-readable score. An automatic process identifies possible problems. Using an interface similar to that of a spelling checker, the user is prompted to audition, accept, or reject each suggested edit. The user can also simply accept all suggestions and listen to the final mix.

While this vision of a highly automated and easy-to-use editing system will require considerable effort to realize, the basic techniques require no fundamental breakthroughs. To illustrate the potential for intelligent, automated editing, and to explore problems that arise in practice, the author has constructed a working prototype that: (1) identifies and labels all notes in a multi-track recording, using a standard MIDI file as a score; (2) estimates the performed pitch and pitch error of each note; (3) applies pitch correction to all notes; (4) locates all simultaneous onsets in the score; (5) aligns recorded notes that are not synchronized within a degree of tolerance; (6) estimates the average recording level for each instrument; and (7) balances the instruments according to user-selected ratios (initially 1:1). The result of this fully automated process is a new set of “corrected” tracks that can be mixed as is, or edited further by hand. While these steps can be performed manually using modern audio editing software, the present work is the *first to automate these steps*, making use of a score (MIDI file) to specify the desired result.

The following sections describe each step of the system, named IAED for *Intelligent Audio Editor*. Section 5 describes some experience with the IAED, and this is followed by a discussion, summary, and conclusions.

2. RELATED WORK

The IAED combines and applies various forms of information derived from audio and scores. Earlier work by the author and others offers many ways to perform the necessary analysis. One of the purposes of the IAED is to gain experience combining these techniques in order to guide further development.

To align tracks to scores, both monophonic and polyphonic score-to-audio alignment techniques are useful. Note-based techniques used in monophonic score following

¹Originally published as: Roger B. Dannenberg. “An Intelligent Multi-Track Audio Editor.” In *Proceedings of the 2007 International Computer Music Conference, Volume II*. San Francisco: The International Computer Music Association, (August 2007), pp. II-89 - 94.

[3] can be applied. There is also work on monophonic and polyphonic alignment [11], [15] that can be used directly. To refine the alignment, onset detection is used. Previous work by Rodet and Jaillet [13], Plumbley, Brossier and Bello [12], Dannenberg and Hu [5], and many others is relevant.

Time alignment and pitch shifting often rely on time stretching techniques that include SMS [16], phase vocoder [8], and PSOLA [14] techniques.

Finally, it should be noted that the practice of fine-grain editing by hand is well established using commercial editors such as ProTools (www.digidesign.com), and plug-ins for time stretching and pitch correction such as Antares Auto-Tune (www.antarestech.com). Non-commercial editors such as Sonic Visualizer [2] and CLAM Annotator [1] offer some assistance labeling pitch and onsets, but are not intended for audio editing.

3. ANALYSIS

The first stage of IAED processing obtains information about each note as well as summary information about the tracks. For simplicity, assume that every track is a recording of a pitched monophonic instrument. Other cases will be discussed later.

3.1. Labeling

The process of identifying notes in the recordings and associating them with notes in the score is called *labeling*. Labeling can be performed in two ways by the IAED. The first method is based on audio-to-score alignment techniques [6]. An alignment program converts both score and audio time frames into sequences of chroma vectors and aligns them using dynamic time warping. [4] This method works for polyphonic recordings as well as monophonic ones. Each track is separately aligned with its corresponding standard MIDI file track, resulting in a mapping between MIDI time and audio time. This mapping is smoothed and used to map each MIDI key-down message to an onset time in the audio recording.

The score alignment is not precise enough for editing, so additional refinement is performed. (Another motivation for the refinement is that we are most interested in notes that occur early or late. Their onset times may not be predicted very well if the score alignment is presented as a smooth tempo curve.) An onset detection function, based mainly on change in the local RMS energy is used to estimate the onset probability of each 5ms frame in the recorded audio. For each note in the score, this probability is weighted by a Gaussian centered on the expected note. This is a roughly Bayesian model where the RMS feature represents an observation probability and the Gaussian represents the prior probability based on the score. The location where their product is maximized determines a refined estimate of the note onset. This is a partial implementation of a

bootstrapping method [5] that proved to work extremely well for onset labeling.

The second method for note onset detection dispenses with the score alignment step and uses labeled beats instead. Beats can be labeled simply by tapping along with the music, or the music might already have a click track or a synchronized MIDI track, in which case the beat locations are already known. The score is warped using linear interpolation between beats. Then, the estimated onset times are refined as described in the previous paragraph.

To verify the labeling, clicks can be synthesized at the note onset times. Then by listening to the rhythm and timing of the clicks, the user can quickly detect any major errors. Figure 1 shows three input tracks and labels (below each audio track) generated automatically by IAED. The number shown by each onset time is the index of the note in the track. Seconds are shown at the very top.

3.2. Dynamics and F0 Estimation

Once note onsets are identified, the IAED labels each note with an indication of dynamic level and fundamental frequency (F0). Since dynamic level is estimated for the purposes of balance between tracks, it is assumed that the subjective impression is created by the loudest part of the overall note. For simplicity, RMS is used rather than a perceptually based measure. The dynamics estimation algorithm, which operates on each note, is as follows:

- Pre-compute an RMS value for each 5ms frame (a by-product of the labeling step described above).
- Find the location of the highest RMS value between the current note onset and the next note onset.
- Scan backward from the peak RMS point until the RMS falls to half the peak value (but stop at the note onset if it is reached). Call this time point A.
- Scan forward from the peak RMS point until the RMS falls to half, stopping if necessary at the next note onset time. Call this time point B.
- Compute the mean RMS over the interval [A, B].

Fundamental frequency is estimated using the YIN algorithm [7]. YIN generates an F0 estimate and a confidence level for each 5ms frame. On the assumption that intonation is most important when notes are loudest, we average the

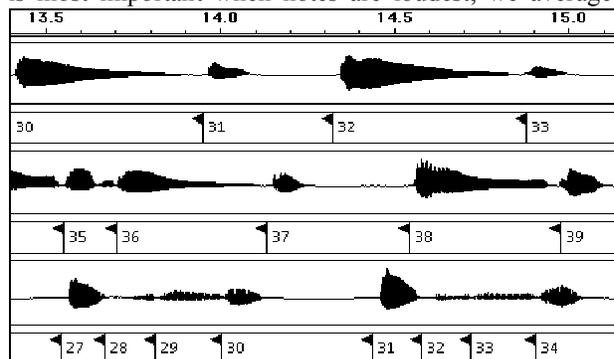


Figure 1. Three tracks (Bb Tpts) with onset labels.

YIN estimates over the same interval $[A, B]$ used to average RMS values, but ignoring frames where the YIN confidence level is low.

3.3. Coincident Notes

The IAED makes many timing decisions based on notes that *should* sound at the same time in different tracks. These coincident notes are detected by scanning the score to find note onsets that match within a small tolerance.

4. EDITING AUTOMATION

After the analysis stage, the IAED computes plans for editing the tracks and then carries out these plans to produce new tracks. In principle, these plans could be reviewed and edited by a user, but the IAED prototype operates in a fully automated batch mode. The editing operations are pitch correction, time alignment, and balancing.

4.1. Pitch Correction

Pitch adjustments are based on comparing each F0 label to the MIDI key number of the corresponding note in the score. A variable-rate, high-quality resampling algorithm, based on sync interpolation [17], is used to warp the original track. Each note is stretched or compressed according to how much sharper or flatter it is relative to the reference note in the score. The idea behind time-varying resampling is to map each sample time in the output to a continuous time in the source signal (a recorded track). The source signal is then interpolated to obtain a value at this time point. This process is repeated to generate each output sample.

The mapping from output sample time to source signal time is computed in several steps because it is simpler to think in terms of mapping from source to output. First, a frequency ratio is computed for each note, representing the amount by which the signal should be stretched or compressed to produce a “correct” pitch. Let ratio $R_i = M_i / \omega_i$, where M_i is the fundamental frequency indicated by the MIDI score for note i , and ω_i is the estimated actual frequency of note i (see Section 3.2). A function is then constructed to represent the time-varying stretch factor to be applied to the track:

$$E(t) = R_i \text{ s.t. } i \text{ satisfies } O_i \leq t < O_{i+1} \quad (1)$$

where O_i is the onset time for note i . $E(t)$ is just a piece-wise constant function with the stretch factors as values. If we integrate $E(t)$, we get a mapping from the original signal times to corresponding time points in the output signal. Taking the inverse, we get a mapping from stretched time to original time. Letting X be the original recording and Y the pitch-corrected signal:

$$Y = X \circ W, \text{ or } Y(t) = X(W(t)), \text{ where} \quad (2)$$

$$W = V^{-1}, \text{ and} \quad (3)$$

$$V(t) = \int E(t) dt \quad (4)$$

In the implementation, E is represented numerically as a function sampled at the frame rate of about 200Hz. This is easily integrated, and since the result is monotonically increasing, the inverse is also easy to compute as another sampled function. The output is just $Y(t)$ computed at discrete sample times. $W(t)$, the “warp” function, is evaluated by linear interpolation to determine a time point in the source signal X . Then $X(W(t))$ is calculated from X using sync interpolation.

As a result of resampling, the note onset times are changed. The note onset labels are changed accordingly from O_i to $W(O_i)$.

4.2. Time Alignment

One of the interesting questions for editing is how to determine what note onset times *should be*. A simple answer is to quantize note onsets to match the times given in the symbolic score, but in most cases, this produces a very mechanical and lifeless result.

Another approach computes the instantaneous tempo of each track as a piece-wise constant function between each pair of adjacent note onsets. If this “tempo curve” is represented as a function from score position (measured in beats) to tempo, then the tempo curves of all tracks can be averaged to form a composite tempo curve. Consider, however, the case where many instruments are either holding notes or not playing while the tempo changes. If tempo curves are constant between note onsets, these instruments will contribute constant values to the average tempo even though the tempo should be changing. A possible solution is to use a weighted average to suppress the influence of instruments that are not establishing the tempo [9]. On the other hand, it is common for musicians to exhibit tempo variation in moving lines due to rhythmic variation or technical difficulties. Overall, the system should find a balance between maintaining a smooth average tempo and allowing for expressive timing variations that can appear as local tempo fluctuations.

A third possibility is to link instruments to a “master” track. For example, three lower voices could be tied to the timing of the top voice in a quartet. The IAED prototype uses this technique. The timing of the lower voices is altered so that coincident notes in the score are coincident in the adjusted audio.

Timing adjustments are computed in several steps. First, the coincident notes between the master track and a “slave” track are identified. For each pair of coincident notes, the onset difference tells how the slave timing should be adjusted. Let (M_i, S_i) be pairs of onset times for coincident notes in the master and slave tracks. Then the sequence $[M_i - S_i \mid 1 \leq i \leq N]$ describes the amounts by which to shift coincident notes in the slave track.

Timing will be altered by time stretching and shrinking sections of music. The i^{th} section spans the time interval $[S_{i-1}, S_i]$. When one section of music is time-stretched, all following notes are affected. Therefore, the amount of time stretching must be decreased by the running total of all previous time stretching:

$$\Delta_i = (M_i - S_i) - \sum_{j=1}^{i-1} \Delta_j \quad (1)$$

We now have a list of audio segments $[S_{i-1}, S_i]$ and the amounts Δ_i by which to lengthen them (negative values mean to shorten them).

Ideally, the stretching or shrinking of segments should be based on a high-quality time-stretching algorithm. For monophonic tracks, one method, based on PSOLA [14], is to identify pitch periods in areas where the signal is highly periodic. Then, duplicate a selected set of these periods to stretch the signal, or delete them to shrink the signal. Because only isolated whole periods are deleted or inserted, this method does not suffer from phase discontinuities or other frame-rate artifacts. Other techniques such as SMS [16] and the phase-vocoder [8] can also be applied.

In the prototype IAED, an even simpler system for time stretching is used. If there are rests in the segment, silence is simply inserted or deleted from the rests. If there are n rests in segment i , each rest is modified by Δ_i/n . (Locating these rests in the actual signal is easy since each rest is terminated by a labeled note onset). If the segment has no rests and is too long, the last note of the segment is truncated and cross-faded with the note-onset of the following segment. If the segment is too short, a bit of silence is inserted at the end of the segment. In general, time adjustments are small, so this simple method works reasonably well.

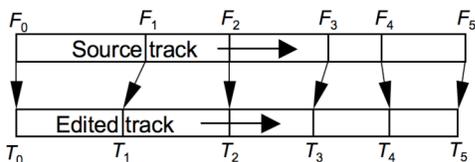


Figure 2. Editing is performed by splitting source tracks on selected note onsets, time stretching, and reassembling to form an edited track.

The signal processing required for time alignment is essentially a matter of cutting out segments of audio from the original track and reassembling them, using short cross-fades to avoid clicks at the splice points. (See Figure 2.) The IAED computes a list of all cutting and pasting operations. Each operation is described by a triple, (T_i, F_i, D_i) , where T_i is the time to play clip i , which is taken from the original track at offset F_i and duration D_i .

To incorporate other forms of time stretching, the same approach can be taken. The essence is to break up each source track into contiguous segments (perhaps with a little overlap to allow cross-fades), time-stretch the segments

using any available technique, and then reassemble the segments to form result tracks.

4.3. Balancing Levels

Track levels are often critical, and setting them requires musical knowledge and at least a good model of human auditory perception. Fine adjustments are left to the human user, but the IAED can at least make an initial rough setting. For each track, the IAED computes L_{avg} , the average of all RMS labels (see Section 3.2). The whole track is then scaled by L_{nom}/L_{avg} , where L_{nom} is the desired, nominal track level. It might be the case that there are desired differences in levels when an instrument is playing solo vs. in ensemble. As an enhancement, IAED computes L_{avg} using only notes that are in the coincident set, i.e. notes where the player should be trying to balance levels with other players. When the adjusted RMS levels of individual coincident notes are not well matched, the interactive version of the IAED can suggest further refinement.

5. RESULTS

The IAED is fully implemented in prototype form. The purpose of the prototype is to test the various components and set the stage for further research and development. So far, it has only been used for small test examples. The evaluation is necessarily subjective, but the results are very encouraging. The results from labeling, pitch correction, time alignment, and balancing levels are described below.

The labeling process generally works well, but editing has very little tolerance for errors. Initial attempts to use score alignment left a few mistakes, primarily where there were repeated notes. To solve this problem, prior research on accurate onset detection [5] will be integrated and tested in this environment. This earlier work achieved 100% accuracy labeling trumpet onsets. It was not integrated into the IAED simply for ease of implementation and the (over-optimistic) assumption that a simpler approach would be adequate.

The alternate labeling method uses taps rather than score alignment for initial note placement. Using this system, there were also a few problems. The original recordings used for testing were made without a click track and had some rather large timing errors. Figure 3 illustrates a section where the top voice, recorded first, has no moving line for 6 beats. When the second track was recorded, the moving line took too long, leading to a proverbial “train wreck” where the parts come together.

Figure 1 begins around the last 2 beats of the music shown in Figure 3. This is an extreme case where there really is no common beat shared by all the parts, and tapping to suddenly shifting time is not an easy job. To resolve this problem, the click track was edited manually to shift a few clicks into synchrony with the music on the track. In a production system, the obvious solution is to provide for

interactive editing, but only where more automated techniques break down.



Figure 3. Fragment from score of test piece (m. 9-11).

Once note onsets and pitches are properly labeled, pitch correction works very well. Even when note onsets are not perfectly detected, the pitch-corrected version sounds very good. Probably the instability of pitch during note transients and the fact that pitch changes are on the order of 1% masks any artifacts that are introduced. Since pitch correction is applied continuously, there are no splices or other forms of editing, and the resulting pitch-corrected signal is continuous, which also minimizes any artifacts.

Time correction introduces the most dramatic changes. Figure 4 shows the output after editing the input shown in Figure 1 (the “train wreck”). Even a quick visual inspection reveals how much better the parts are aligned. While some extreme changes are made in the test tracks, and considering that very simple time-stretching techniques are used, the result is quite good. The fact that edits are made only milliseconds before coincident attacks undoubtedly helps to mitigate any signal processing artifacts. There do seem to be some musical artifacts: Even though time correction only changes note durations by tens of milliseconds, this can be rhythmically important. Better time stretching that distributes the stretch evenly over segments (which may range from one to many notes) should help in this area.

In the test data, the average levels, L_{avg} , of all parts were approximately equal, so track level adjustments were insignificant. However, the resulting balance is acceptable, indicating that this is at least a plausible way to automatically set initial levels.

As important as it is to “correct” the original performance material, it is crucial to retain its desirable musical and expressive attributes. Since pitch is not corrected within notes, expressive and characteristic pitch changes are retained and sound natural. Since levels are not adjusted between notes but only between tracks, expressive dynamics are also retained. The one area where expressiveness seems to be compromised is the rhythmic “feel.” This is an inherent problem even with manual editing, and overall, the IAED processing improves the recording. As automatic editing moves from research to practice, there will be opportunities to identify the sources of rhythmic “feel” and to better retain this “feel” even while correcting timing.

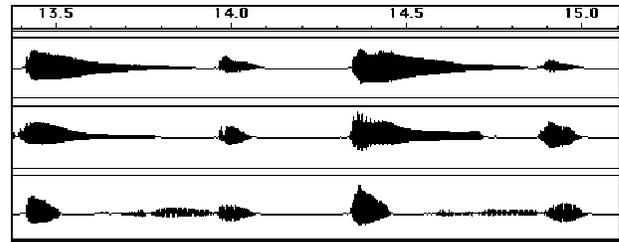


Figure 4. Output generated by the IAED.

6. DISCUSSION

The results are quite encouraging. IAED can start with a rather sloppy multi-tracked performance, a MIDI version of the score, and some tapped beat tracks. Without further intervention, IAED generates an edited version with better intonation, greatly improved rhythmic accuracy, and a good initial balance. The result retains much of the musicality and expressivity of the original tracks.

The prototype has revealed many interesting issues, setting the stage for future research and development. One of the most interesting issues is tempo and beat placement. There are a variety of sometimes-conflicting goals, including:

- Avoid long-term tempo changes.
- Allow slight tempo changes for expression.
- Avoid phrases that rush or drag.
- Make rhythms accurate.
- Allow expressive timing deviations, e.g. for swing.
- Align note onsets to the expected onset times.
- Align note onsets to those in other tracks.

If tempo is viewed as a signal, we can think of filtering the tempo to achieve different effects. A high-pass filter allows local timing deviations and eliminates long-term changes. A low-pass filter eliminates local time deformations, but admits longer-term tempo changes. It might be advantageous to have an interface something like a graphical equalizer to control tempo variation on different time scales.

Determining how best to adjust note timing is an interesting area for future work. A good illustration of the subtlety of this problem is measure 2 of staff 2 in Figure 3. These notes should be aligned to the *implied* beat from measure 1 of staff 1, even though there are no shared onset times. This example illustrates the tension between the opposing goals of steady tempo and expressive performance timing.

It is tempting to synchronize every pair of coincident notes, but systematic deviations from synchrony can be musically important. Sometimes melody leads the accompaniment, perhaps to make the melody more salient, but musicians also “lay back,” placing melodic onsets later in time than the accompaniment. The best way to handle this may be to give the user the ability to apply time correction

selectively. Moreover, the very notion of aligning note onsets is only an approximation to a more sophisticated view that considers perceptual onset times as well as timbral and musical implications of onset timing. Perhaps desired onset timings can be learned through a statistical analysis of the performance. Assuming that onset timing is good for the most part, it is possible to detect and “correct” outliers to match the mean rather than simply line up all onsets.

Pitch correction is another interesting area for future research. The IAED simply adjusts pitches to equal temperament, but future systems might use an automatic harmonic analysis of the score [20] to achieve better intonation. For example, major thirds could be tuned to a 5/4 ratio, based on “just” intonation, which is substantially smaller than an equal-tempered third. The IAED uses a particular model to predict perceived pitch, but now that the problem is posed, surely future research will develop better models. Finally, the IAED intentionally does not modify pitch within a note, but this is easily done by changing $E(t)$ in Equation 1. This could be used to manipulate vibrato, pitch change during the attack, and other pitch variation

IAED does not adjust note durations or the dynamics within a note. In the test data, there are some crescendos that are not perfectly matched. One could simply adjust the RMS envelope of one note and the cutoff time to match another’s. This is just one example of how articulation might be altered to improve the ensemble.

These techniques may not apply very well to polyphonic instruments such as piano, or to non-pitched instruments, e.g. drum sets. However, onset detection and alignment might still be possible if the editing decisions are made manually. The user interface and support for manipulating these sorts of tracks are left to future work.

The next step will be to process music recorded by a jazz octet. These octet recordings include a large amount of notated music for three saxophones, trumpet, and trombone, while the piano, bass, and drums are largely unnotated. The arrangements were created using a music notation editor that can save standard MIDI files. Therefore, IAED can read the scores and operate on the horn tracks.

7. CONCLUSIONS

Detailed editing of multi-track recordings is a standard practice in modern music production. Editing is used to perfect professional recordings and, with perhaps more work, editing can make amateur recordings sound professional. For the most part, editing operations are quite mechanical and systematic. Using state-of-the-art techniques for audio analysis and processing, the Intelligent Audio Editor, or IAED, automates these tedious operations. IAED uses a score to help analyze the audio and also to serve as a specification for the desired music. Edits correct pitch errors, timing errors, and overall signal imbalance while retaining most of the expressive nuance, including

expressive timing, articulation, and pitch variation such as vibrato. Many interesting issues are raised in this work, suggesting future research on score-to-audio alignment, onset detection, performance timing, and perceived pitch. Additional work may also be needed on interfaces that integrate audio with music notation and allow users to request, constrain, and adjust “intelligent” editing operations.

8. REFERENCES

- [1] Amtrian, X., J. Massaguer, D. Garcia, and I. Mosquera. “The CLAM Annotator: A Cross-platform Audio Descriptors Editing Tool.” In *Proceedings of 6th International Conference on Music Information Retrieval*. London: Queen Mary, University of London, 2005, pp. 426-9.
- [2] Cannam, C., C. Landone, M. Sandler, J. P. Bello. “The Sonic Visualizer: A Visualization Platform for Semantic Descriptors from Musical Signals.” In *ISMIR 2006 7th International Conference on Music Information Retrieval Proceedings*. Victoria, BC, Canada: University of Victoria, 2006, pp. 324-7.
- [3] Dannenberg, R. “Real-Time Scheduling and Computer Accompaniment.” In Mathews, M. and J. Pierce, eds. *Current Research in Computer Music*. Cambridge, MIT Press, 1989.
- [4] Dannenberg, R. and Hu, N. “Polyphonic Audio Matching for Score Following and Intelligent Audio Editors,” in *Proceedings of the 2003 International Computer Music Conference*, San Francisco: International Computer Music Association, 2003, pp. 27-34.
- [5] Dannenberg, R. and Hu, N. “Bootstrap Learning for Accurate Onset Detection,” *Machine Learning* 65:2-3, December 2006, pp. 457-471.
- [6] Dannenberg, R. and Raphael, C. “Music Score Alignment and Computer Accompaniment,” *Communications of the ACM* 49:8, August 2006, pp 38-43.
- [7] de Cheveigné, A. and Kawahara, H. “Yin, A Fundamental Frequency Estimator for Speech and Music.” *Journal of the Acoustical Society of America*, 111(4), 2002.
- [8] Dolson, M. “The Phase Vocoder: A Tutorial.” *Computer Music Journal* 10(4), 1987, pp. 14-27.
- [9] Grubb, L., and R. B. Dannenberg. “Automated Accompaniment of Musical Ensembles.” *Proceedings of the Twelfth National Conference on Artificial Intelligence*. AAAI, 1994, pp. 94-99.
- [10] Hu, N., R. B. Dannenberg, and G. Tzanetakis. “Polyphonic Audio Matching and Alignment for Music Retrieval.” In *2003 IEEE Workshop on Applications of Signal Processing to Audio and*

- Acoustics (WASPAA)*. New York: IEEE, 2003, pp. 185-188.
- [11] Orio, N. and Schwarz, D. "Alignment of Monophonic and Polyphonic Music to a Score," in *Proceedings of the 2001 ICMC*. San Francisco: International Computer Music Association, 2001, pp. 155-158.
- [12] Plumbley, M., Brossier, P., and Bello, J. "Fast labelling of notes in music signals." In *ISMIR 2004 Fifth International Conference on Music Information Retrieval Proceedings*. Barcelona, Spain: Universitat Pompeu Fabra, 2004, pp. 331-6.
- [13] Rodet, X., and F. Jaillet. "Detection and Modeling of Fast Attack Transients." *Proceedings of the 2001 International Computer Music Conference*. International Computer Music Association, 2001, pp. 30-33.
- [14] Schnell, N., Peeters, G., Lemouton, S., Manoury P., Rodet, X. "Synthesizing a Choir in Real-Time Using Pitch Synchronous Overlap Add (PSOLA)", In *Proceedings of the International Computer Music Conference 2000*, San Francisco: International Computer Music Association, 2000.
- [15] Schwarz, D., Cont, A., and Schnell, N. "From Boulez to Ballads: Training IRCAM's Score Follower," in *Proceedings of International Computer Music Conference*. San Francisco: International Computer Music Association, 2005.
- [16] Serra, X. and Smith, J. Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic Plus Stochastic Decomposition. *Computer Music Journal*, 14(4), 1990, pp. 12-24.
- [17] Smith, J. and Gossett, P. "A Flexible Sampling-Rate Conversion Method," In *ICASSP'84, 19.4.1-19.4.4*. San Diego, CA, March 1984, <http://www-crma.stanford.edu/jos/resample>.
- [18] Soulez, F., X. Rodet, and D. Schwarz. "Improving Polyphonic and Poly-Instrumental Music To Score Alignment." In *ISMIR 2003 Proceedings of the Fourth International Conference on Music Information Retrieval*. Baltimore: Johns Hopkins University, 2003, pp. 143-150.
- [19] Spiegl, S. "Three for the Money" in *Jazz Inventions: Duets, Trios for Trumpet*. N. Hollywood Maggio Music Press, pp. 28-34.
- [20] Temperley, D. *The Cognition of Basic Musical Structures*. MIT Press, 2001.