The Intelligent Music Editor: Towards an Automated Platform for Music Analysis and Editing

Yuxiang Liu[1], Roger B. Dannenberg[2], Lianhong Cai[1]

[1] Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing, China

[2] School of Computer Science, Carnegie Mellon University, Pittsburgh, USA

liuyuxiang06@mails.tsinghua.edu.cn, rbd@cs.cmu.edu, clh-dcs@tsinghua.edu.cn

Abstract. Digital music editing is a standard process in music production for correcting mistakes and enhancing quality, but this is tedious and time-consuming. The Intelligent Music Editor, or IMED, automates routine music editing tasks using advanced techniques for music transcription (especially score alignment), and signal processing. The IMED starts with multiple recorded tracks and a detailed score that specifies all of the notes to be played. A transcription algorithm locates notes in the recording and identifies their pitch. A scheduling model tracks instantaneous tempo of the recorded performance and determines adjusted timings for output tracks. A time-domain pitch modification/time stretching algorithm performs pitch correction and time adjustment. An empirical evaluation on a multi-track recording illustrates the proposed algorithms achieve an onset detection accuracy of 87% and a detailed subjective evaluation shows that the IMED improves pitch and timing accuracy while retaining the expressive nuance of the original recording.

Keywords: Intelligent Music Editor, Music Transcription, Score-Audio Alignment, Pitch Estimation, Time Stretching, Pitch Shifting.

1 Introduction

Editing allows recording engineers and producers to make *incremental* changes to audio rather than discarding a mostly-good recording and starting over. Ultimately, the biggest limitation of editing is the human time it takes to perform the edits. Since most edits simply adjust notes to achieve better rhythmic and tuning accuracy, it seems quite possible to automate a large fraction of the most desirable edits. This could lower recording costs and enable many more creative musicians to produce recordings with a professional sound.

In this paper, we describe an Intelligent Music Editor (IMED) and discuss problems that arise in practice. As a highly automated and easy to use system for music analysis and editing, IMED is able to analyze music content by linking signal and symbolic representations of music. The overall strategy is to use symbolic music descriptions (MIDI, for example) as a specification for the music. This specification is compared automatically to a multi-track recording on an instrument-by-instrument basis to determine deviations in the performance. As for editing automation, IMED can automatically manipulate music recordings, according to an automatically generated plan or user instructions, by moving notes, stretching time, correcting pitch, and mixing tracks. The output of the system is an edited version of the original with adjustments to timing, tempo, pitch, and loudness.

On one hand, IMED offers great flexibility to musicians and editors by allowing recordings to be automatically refined in term of pitch, timing and dynamic level. On the other hand, content-based music analysis is useful for music understanding and retrieval. Currently, music structure labeling is performed manually. Hence, IMED's analysis techniques can effectively reduce the workload of human annotators and would be an indispensable component of a music information retrieval system.

2 Related Work

An early work describing the need of intelligent audio editor was presented by Chafe [1]. Although the concept that an audio editor should be able to make use of music content was promising, the actual system was not very practical due to technical limits of that time.

Tzanetakis [2] implemented a working prototype of an intelligent editor for Jazz music, which provided an interactive experimentation environment for combining and testing content-based analysis components in the domain of Music Information Retrieval (MIR).

A number of studies on score-audio alignment [3-5], structural analysis [6,7] and analysis-resynthesis of signals [8] have been performed. However, there has been little work on integrating these techniques into a platform except the former prototype of IMED [9]. This early work was in essence a proof-of-concept and not intended as for production work. The present study is the first attempt to use large-scale automatic editing techniques on an actual studio recording. This study raises many practical problems that did not arise in earlier work, which often looked at carefully selected data.

3 Analysis: score-assisted music transcription

The major task of the analysis stage is to transcribe performed music into a corresponding score. In another words, the program identify notes in an audio track and associate them with notes in the reference score. A large number of automatic techniques for music transcription have been proposed，the majority of which extract notes directly from music signals and seldom involve reference scores. In our case, the full transcription is not necessary because we have a reference score and we expect the performers to adhere closely to the score. While music transcription is still a largely unsolved problem, audio-score alignment is relatively easy. In IMED, this is accomplished in the two steps: score-assisted high-resolution onset detection and pitch estimation.

3.1 Accurate onset detection

Our problem of onset detection can be summarized by the following expression:

$$\operatorname{argmax} P(T \mid R, M)$$

We want to find the best (most likely) correspondence T between a given audio recording, *R*, and score, *M*. The recording comes from a real-time performance, so it will naturally contain deviations from a precise rendering of the score. On the other hand, the score is an abstract, discrete representation of a continuous phenomenon, and composers expect and rely upon musicians to interpret the score with expression and feeling, so there will always be deviations.

To form a correspondence between R and M, we first apply an overall global alignment to stay on track and then use some more refined methods at the note-by-note level in order to achieve a high temporal accuracy.

Score Alignment

The essence of audio-score alignment is as follows: Audio signals are divided into 50ms long, half-overlapping frames. Both score and audio frames are converted into sequences of chroma vectors [10]. A distance metric is used to measure dissimilarity between these chroma vectors. Finally, we employ Dynamic Time Warping (DTW) to find the optimal match between the two time series. Please refer to our previous work [4] for more detail. Each track of the multi-track recording is separately aligned to the corresponding part (a MIDI track or channel) in a reference score.

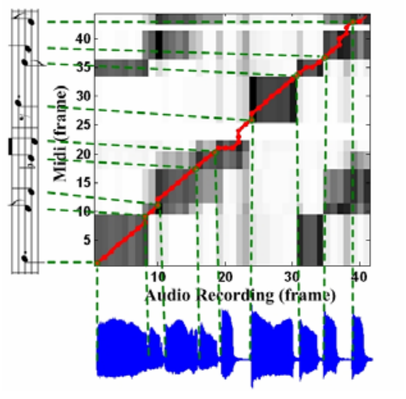

Fig. 1 Audio-Score Alignment

In an ensemble, not all musicians play all the time. Therefore, we need a silence detection process to determine segments where the musician is actually playing. Usually, this can be achieved by tracking energy of sounds and using an empirical threshold to distinguish segments with low energy. Unfortunately, in most cases, especially for a live concert, musicians cannot be separated perfectly from each other and the interference among instruments is inevitable. Hence even when a musician does not play anything, other sounds appear on the track and it is hard to estimate a universal energy threshold. To avoid this difficulty, we apply a silence alignment procedure to estimate dynamic level of noise. Our silence alignment procedure is basically the same with score alignment, but we replace the chroma vector with Short Time Energy (STE) and redefine the distance measure as below:

$$w \times | STE_{mid} - STE_{audio} | + (1-w) \times | d(STE_{mid}) - d(STE_{audio}) |$$

where the first term is difference of STE between a midi frame and an audio frame. The second term is difference of the first derivative of STE, which reflects dynamic difference between midi and audio. This definition is motivated by the phenomenon that for any individual note generated by a musical instrument, there is an attack part. An attack is extremely salient when it occurs right after a silent segment and makes a rapid amplitude rise in the waveform. We believe involving dynamic difference of STE is useful for detecting silence boundaries.

A pitch change will definitely result in a sudden spectral change. Inspired by this observation, when computing dissimilarity between midi and audio frames, we take spectral changes between successive frames into account so as to improve temporal accuracy at note boundaries. In our implementation, the overall distance is defined as:

$$w \times Dist(M_i, R_j) + (1-w) \times | Dist(M_i, M_{i-1}) - Dist(R_j, R_{j-1}) |$$

where M_i stands for the chroma vector of the i^{th} frame in a Midi, and R_j is the chroma vector of the j^{th} frame in a audio recording. $Dist(a,b)$ is Euclidean Distance between vector a and b. The first term corresponds to the spectral difference between midi frame and audio frame, while the second term can be considered as the difference of their first derivative.

Once the distance measure is specified, the rest of the process is essentially a matter of searching for a shortest path through the dissimilarity matrix using dynamic programming.

Bootstrap learning for accurate onset detection

Due to the limitation of chrome feature extraction, the analysis windows size cannot be shorter. Hence the temporal resolution of score alignment is 25ms in our implementation. In our experience, even highly overlapped windows with a smaller hop size do not improve the temporal resolution of alignment results. However, 25ms is not precise enough for editing, so additional refinement is needed. To this end, we use alignment data to train an onset classifier.

The features used for onset classification are energy, fundamental frequency, the relative strengths and frequency deviations of the first three harmonics, and the zero-crossing rate. We use overlapping analysis windows of size 23.2ms (1024 samples at a sample rate of 44.1 kHz). The hop size is 5.8ms (256 samples), providing a high temporal resolution.

Typically, onset detection systems are intended to work with a wide variety of inputs, and for detectors based on machine learning, one would expect to require a large set of hand-labeled training examples. In our system, however, it is an advantage to fit the detector to a particular instrument or even a particular performer. Furthermore, we can use alignment data rather than hand-labeled data as training data. In our previous work [11], we describe a semi-supervised, or bootstrap learning approach where we train a classifier using alignment data. Then the classifier is used to re-label onsets, improving the alignment. This process is iterated until it converges, giving an accurate set of onset labels as a side-effect of training the classifier. Due to space limitations, we refer the reader to a previous publication [11] for details.

3.2 Pitch Estimation

Once musical signals are segmented into notes, the YIN algorithm [12] is used to estimate a pitch for each note. In YIN, an average magnitude difference function (AMDF) between a segment of the signal and the same segment lagged by a trial period is calculated. (Shown in Fig. 2(a)) The algorithm searches for a minimum (Point P2 in Fig. 2(a)) throughout the AMDF curve, which varies as a function of lag.

Although this approach works well, like most pitch estimation algorithms, it suffers from too low/high errors, where a longer period (P3 in Fig. 3a) or a shorter period (P1 in Fig. 2(a)) valley is chosen incorrectly. These valleys often occur an octave higher or lower than actual pitch.

In a music performance, it is rare that a performer plays a note that is an octave away from a reference note in a score. This encourages us to restrict the search range in a small neighborhood around the reference pitch (Fig. 2(b)). Our experiments show that this simple method works extremely well for eliminating octave errors.

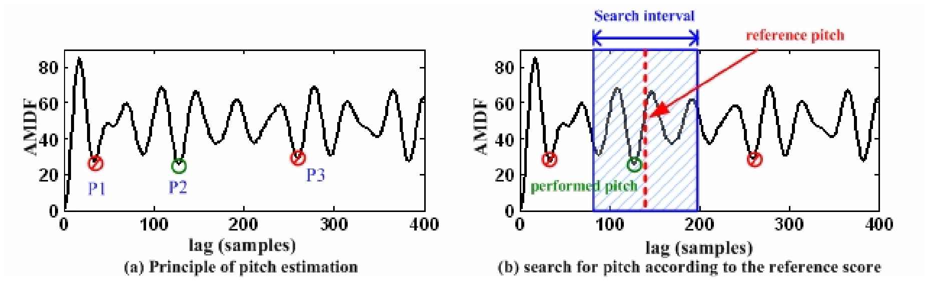

Fig. 2 Pitch Estimation Algorithm

4 Automatic Editing

Usually, performed tracks differ from both the reference scores and each other in terms of note timing, durations, and tuning. This is due to differences in music interpretation, limitations to human accuracy, and simply performance errors. In order to make tracks sound more natural and coherent, the IMED first constructs plans to alter the labeled notes in terms of starting time, duration, and pitch.

4.1 Reschedule Timing and Determine Pitch

At first, it might appear obvious how to modify note timing – simply adjust the time to match the time in the score. This simple approach might be useful, but in general it will take out the “life” of the music, making it sound mechanical or “robotic.” What we want is to capture the subtle and expressive timing variation of the human performers while at the same time reducing the timing differences that simply reflect unintentional early or late entrances and other technical difficulties.

IMED must therefore track the overall tempo of the whole ensemble and then adjust each individual instrument track to match the group. We assume that tempo is relatively stable and drastic tempo changes do not happen. Accordingly, we can estimate instantaneous tempo at a reference score position by linear regression from actual performed onset times nearby.

To calculate the instantaneous tempo at time *x* in a midi track, we pick up notes, whose onset (based on midi time) is in the sliding window [*x*-T/2, *x*+T/2], from all tracks. The sliding window size T is set to 20 seconds empirically, which allows the linear regression procedure to span about forty beats. A linear regression procedure is then applied to find the least squares fit of a linear function that maps beat to time. The predicted onset time for beat *x* is simply the value of the fitted function evaluated at *x*.

Compared to timing, pitch determination is rather simple. A number of performance practices will be used by musicians for expression, such vibrato, glissando and portamento, during a performance. A good editor is required to retain these expressive effects. Therefore, instead flattening performed pitch to the pitch corresponding to the MIDI number in the score, we shift the whole pitch curve by an interval between the weighted average of the estimated pitch curve and the pitch derived from MIDI key number.

4.2 Time Adjustment and Pitch Shifting

Time stretching and pitch shifting are carried out simultaneously by a high quality timescale-pitch modification algorithm based on Pitch Synchronous Overlap and Add (PSOLA) [13]. Our implementation relies on the élastique SOLOIST 2.0 SDK by zplane.development [14].

In order to avoid clicks at splice points, the whole track is edited in a continuous manner. Thus, if there is a phrase with several notes, we do not separate the notes, transform them, and splice them back together. Instead, we transform all of the notes using time-varying stretch factors and pitch shift amounts, allowing the PSOLA algorithm to handle the details.

Consider, however, that PSOLA is by definition pitch synchronous, so only whole periods can be inserted or deleted to change the duration of a note or segment of audio. This means the length of output signals is not guaranteed to satisfy the requested stretch ratio exactly. Although differences are rather small respectively (not larger than one period each), accumulated errors could still affect quality of result tracks.

To avoid accumulated quantization error, we update the stretch ratio for each note iteratively, treating the PSOLA algorithm as a “black box” whose next input comes from the *i*th sample in the source track and whose next output will be written to the *j*th sample of the destination track. Now, suppose when the program begins to process the *k*th note that the next note (at *k*+1) has onset times corresponding to samples *i'* and *j'* in the source and destination tracks, respectively. The stretch ratio for the *k*th note should then be $\frac{j' - j}{i' - i}$, which will place the *k*+1th note as accurately as possible, independent of any previous quantization error.

5 Evaluation

Although we have been working with many pieces of music, a detailed evaluation has been carried out on a recording of “Embraceable You”. The music lasts 3 minutes and 45 seconds. The instrumentation consists of five “horn” tracks: alto saxophone, tenor saxophone, baritone saxophone, trumpet and trombone. The total number of notes in all five tracks is 1767. The performance was recorded in a studio, and all musicians played at the same time. There was a close microphone in front of each musician. There is obvious “cross-talk” between channels, but the multi-track source material was edited extensively by hand for a resulting compact disc.

To measure the performance of the onset detection algorithms which include silence alignment, note alignment, and bootstrapped onset detection, we run the detector on the acoustic recordings and then manually correct mistakes by moving wrong onset times to appropriate positions in the recording. Because the detector is based on score alignment, extra or missing note errors occur only when a performer played incorrectly. Other errors include inaccurate onset timing and note shifting, where onsets are correctly identified but assigned to the wrong note in the score. Because revising all the note onsets in recordings is too time consuming, we only correct obvious mistakes whose deviation is larger than 25ms.

Tab. 1 Accuracy of onset detection

	Trumpet	Alto Sax	**Tenor Sax**	**Baritone sax**	Trombone	Overall
Correct Onset	316	355	357	238	277	1543
Total Onset	326	371	373	326	361	1767
Accuracy	96.93%	95.69%	95.71%	73.00%	76.73%	87.32%

Tab. 1 illustrates the overall detection accuracy. As shown, the overall accuracy is 87%. For trumpet, alto sax and tenor sax, the accuracy is much better, with all above 95%, showing the feasibility of annotating the music in a fully automatic fashion. However, the performance is not so satisfactory when IMED deals with baritone sax and trombone tracks. The poorer performance could be due to bass characteristics of these two instruments. Take the baritone sax track as an example. Most notes are around pitch C3 (a fundament frequency of 123Hz). Such low pitches prevent the program from extracting spectrum and F0 of audio signals accurately. Although a longer window size for spectral analysis may help to gain a better spectral representation, it will result in a low temporal resolution. Balancing the trade off between spectral and temporal accuracy and improving onset detection in the bass register is an interesting challenge for future investigation.

To compare the edited sound with the original recording, we conducted a subjective evaluation. We have three versions of the recording: an original version without any editing, an automatic edited version without any manual intervention, and an edited version based on handmade corrected onset labels. For each version, all instruments are mixed into one track. A subject first listens to the original version picking up problems where notes are not synchronized in the recording or where a note is played out of tune. Then he listens to the two edited versions and finds out whether errors are fixed. At the same time, he should pay attention to whether the edited sounds are natural enough and retain expressive musical articulations, such as glissando, legato and so forth. Any additional errors in the edited versions are noted.

A total of 58 problems were found in the original recording. There were 42 timing problems, 15 intonation (tuning) problems, and 1 note held too long. IMED improved 33 of 42 timing problems, 3 remained the same, and 6 became worse. In addition, 11 new timing problems were introduced. Of the intonation problems, 8 were improved, 5 were rated the same, and 2 became worse. The single long note problem sounded worse after editing. Finally, there were 3 objectionable editing artifacts detected.

We can say that IMED reduced the original 58 problems, to only 28 problems, cutting the manual editing task by more than half. With hand-corrected timing, the main change is that the number of timing problems that grew worse was reduced from 6 to 3. These numbers are encouraging, but they do not tell the whole story. Overall, the edited recording suffers from a loss of "crispness," probably due to the fact that the main pitch- and time-shifted signals are added to the "bleed through" signals originating from the same instrument but arriving through other tracks, which are processed differently. This creates a chorus-like effect that generally sounds muddy and at times creates noticeable interference sounds. There are several solutions to this, including better isolation of instruments in the recording process, recording instruments one-at-a-time, automatic attenuation of "bleed through" when the main instrument on the track is not playing, and noise removal techniques.

6 Conclusions and Future Work

In this paper, an intelligent music editor, which transcribes music recordings and makes adjustments to note pitch and timing in an automatic fashion, is presented. We believe this represents the first attempt to edit a multi-track, studio recording automatically. By combining score-audio alignment with a bootstrap learning approach, the proposed transcription model yields an overall onset detection accuracy of 87%, which shows the feasibility of a fully automatic music transcription system. A time domain algorithm based on PSOLA is proven effective at pitch shifting and time stretching to achieve a natural and musical result. A subjective evaluation demonstrated that the system automatically corrects pitch errors and adjusts timing without destroying the musicality of the recording. In fact, the process improves the musical quality of the recording as if it were edited by hand.

Nevertheless, there is plenty of room for improvement in the editor. By working with real studio recordings, our work has revealed a number of practical issues that may guide future researchers who are interested in this new problem.

As for score alignment, our current model based on chroma features suffers when pitches are low. A series of time domain features should be considered, and their effectiveness will be explored in our future research. In addition, it is tempting to align all tracks simultaneity, which helps to use timing information of coincident notes in other track when the detector fails to identify an onset in a bass track. Currently, IMED assumes performances are correct except for small timing and pitch errors, so it does not detect outright mistakes such as a missing or extra note. It should be possible to detect mistakes automatically and even use similar notes in the recording to synthesize a performance. For example, string matching algorithms can be used to detect minimal edits to "repair" a note sequence, and performance error detection has already been explored in music education systems.

Acknowledgments

A major part of work was done while the first author was at Carnegie Mellon University. The first author was supported by the National Natural Science Foundation of China (No. 90820304). This material is also based on work by the second author supported by the (U.S.) National Science Foundation Grant No. 0855958. We would like to thank Tim Flohrer, Alexander Lerch, and zplane.development for their élastique SOLOIST 2.0 SDK library.

References

1. Chafe, C., Mont-Reynaud, B., Rush, L.: Toward an Intelligent Editor for Digital Audio: Recognition of Musical Constructs. Computer Music Journal, vol. 6, (1982)
2. Tzanetakis, G., Hu, N., Dannenberg, R.: Toward an Intelligent Editor for Jazz Music. In: IEEE Workshop on Image Analysis for Multimedia Interactive Systems, London, UK, (2002)
3. Orio, N., Schwarz, D.: Alignment of Monophonic and Polyphonic Music to a Score. In: the International Computer Music Conference, San Francisco: International Computer Music Association, pp. 155-158, (2001).
4. Dannenberg, R., Hu, N.: Polyphonic Audio Matching for Score Following and Intelligent Audio Editors. In: the International Computer Music Conference, San Francisco: International Computer Music Association, (2003)
5. Ewert, S., Muller, M., Grosche, P.: High resolution audio synchronization using chroma onset features. In: the IEEE International Conference on Acoustics, Speech, and Signal Processing, Taipei, Taiwan, (2009)
6. Tzanetakis G., Cook, P.: Multifeature Audio Segmentation for Browsing and Annotation. In: Workshop on Applications of Signal Processing to Audio and Acoustics, New Paltz, USA, (1999)
7. Aucouturier J., Sandler, M.: Segmentation of Musical Signals using Hidden Markov Models. In: Audio Engineering Society (AES) Convention, Amsterdam, Netherlands, (2001)
8. Woodruff, J., Pardo, B., Dannenberg, R.: Remixing Stereo Music with Score-Informed Source Separation. In: 7th International Conference on Music Information Retrieval Proceedings, Victoria, Canada, pp. 314-319, (2006)
9. Dannenberg R.: An Intelligent Multi-Track Audio Editor. In: the International Computer Music Conference, San Francisco: The International Computer Music Association, (2007)
10. Fujishima, T.: Realtime chord recognition of musical sound: a system using common lisp music. International Computer Music Conference, Beijing, China, (1999)
11. Hu, N., Dannenberg, R.: Bootstrap learning for accurate onset detection. Machine Learning, vol. 65:2-3, pp. 457-471, (2006)
12. de Cheveigné, A., Kawahara, H.: YIN, a fundamental frequency estimator for speech and music. The Journal of the Acoustical Society of America, vol. 111, pp. 1917-1930, (2002)
13. Moulines, E., Charpentier, F.: Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. Speech Communication, Vol. 9, (1990)
14. Flohrer, T.: SDK Manual (for élastique SOLOIST 2.0 SDK), Berlin: zplane.development, (2007)