INSTLISTENER: AN EXPRESSIVE PARAMETER ESTIMATION SYSTEM IMITATING HUMAN PERFORMANCES OF MONOPHONIC MUSICAL INSTRUMENTS

Zhengshan Shi

Center for Computer Research
in Music and Acoustics (CCRMA)
Stanford, CA, USA
kittyshi@ccrma.stanford.edu

Tomoyasu Nakano, Masataka Goto

National Institute of Advanced
Industrial Science and Technology (AIST)
Tsukuba, Ibaraki, Japan
{t.nakano, m.goto}@aist.go.jp

ABSTRACT

We present InstListener, a system that takes an expressive monophonic solo instrument performance by a human performer as the input and imitates its audio recordings by using an existing MIDI (Musical Instrument Digital Interface) synthesizer. It automatically analyzes the input and estimates, for each musical note, expressive performance parameters such as the timing, duration, discrete semitone-level pitch, amplitude, continuous pitch contour, and continuous amplitude contour. The system uses an iterative process to estimate and update those parameters by analyzing both the input and output of the system so that the output from the MIDI synthesizer can be similar enough to the input. Our evaluation results showed that the iterative parameter estimation improved the accuracy of imitating of the input performance and thus increased the naturalness and expressiveness of the output performance.

Index Terms— performance imitation, expressive musical performance, iterative parameter estimation, performance synthesis by analysis, musical expression

1. INTRODUCTION AND MOTIVATION

Human musical performances are expressive, making musical performances attractive. People tend to refer to performances without any expressions as "robotic" or "deadpan". Even when a computer is used in generating music, people often tend to prefer more expressive performances. Thus researchers have been putting efforts into analyzing and modeling expressive music [1–5]. Pioneers such as Lejaren Hiller and Iannis Xenakis [6] gained access to computers to make music with a "human feel". Since 1957, when Max Mathews first made sound from the computer [7], there have been existing lots of research efforts onto mechanical or computational modeling of expressive performance of music [5, 8]. It has shown that changing the tempo and loudness, and use of expressive articulation are the two most common approaches for expressive performances [9]. Other efforts have been putting onto the structures and phrasing of music, or relationship between pitch and velocity [10, 11].

On the other hand, various works have been done on synthesizing expressive musical performances, for example, rule-based model [12–14], statistical analysis and stochastic model [15, 16], getting physical measurement of musical performance through musical instruments [17, 18], and among others [19–23]. There are also many researchers working on automatic music transcription that aims to accurately transcribe audio performances into score. However, few researches have been working on parameterizing musical expressions other than observing the expressions [24, 25].

Our aim is to fill in the gap by bringing a new insight into the process of generating expressive musical performances. Although musical score is a compact and neat way to imply musical expression, it is not enough to carry all the nuances in a musical performances. For performers and musicologists who are interested in studying acoustic musical performances, an automatic transcription of a performance back to music notation is clearly not enough. We therefore imitate existing musical performances to obtain their faithful reproduction by using MIDI synthesizers. By estimating and controlling continuous expressive MIDI parameters, we will hopefully have a better understanding of musical expressions themselves. Parametric representation helps decoding the mystery of an expressive musical performance into a set of parameters. Thus we consider it useful to make a realistic imitation of the acoustic instrumental performance. It can provide invaluable resource for not only people who study musical performances but also people who apply and transfer certain musical expressions into other domains.

We propose InstListener, a system that analyzes a recording of a musical instrument performance to faithfully transcribe its nuance by reducing expressive performances into several dimensions that can be encoded into MIDI information. The goal of this system is to convert the original input recording into an expressive musical performance in MIDI format that approximates the input well. For the purpose of this paper, we focus on monophonic instruments such as saxophone or clarinet.

2. INSTLISTENER

The InstListener system takes, as the input, an audio file that contains a monaural recording of a monophonic solo instrument performance by a human performer. After a MIDI synthesizer is specified, it analyzes the input and generates, as the output, a MIDI file that contains MIDI notes and parameters for imitating the input performance by using the specified MIDI synthesizer. The system analyzes the pitch contour, the onset time, and the root-mean-square energy (RMSE) for each musical note of the input performance, and then by using those acoustic features (analyzed results), it estimates MIDI parameters of each musical note: the timing and duration (i.e., onset and offset of the note), the discrete semitone-level pitch (MIDI note number of the note), the amplitude (MIDI velocity of the note), continuous pitch control parameters (MIDI pitch bend control), and continuous amplitude control parameters (MIDI volume control).

Since different MIDI synthesizers have different characteristics and expressiveness, the resulting MIDI file should depend on the specified target synthesizer to accurately reproduce the input performance. InstListener therefore leverages an iterative parameter

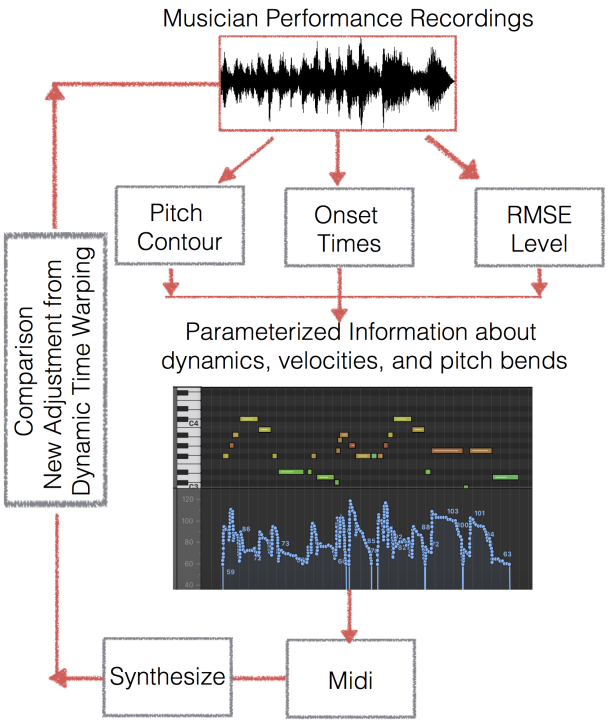

Fig. 1: System workflow of InstListener. The system extracts pitch contour, onset time, and root-mean-square energy (RMSE).

estimation technique, which was proposed by Nakano and Goto in VocaListener [26, 27], that imitates singing voices by generating singing synthesis parameters. It is used as a basis and inspiration of this work.

Even if we provide the same MIDI parameters, different MIDI synthesizers generate sounds having slightly different expressions. InstListener therefore analyzes not only the input, but also the output from the target MIDI synthesizer in the same manner, and then compares their acoustic features. On the basis of this comparison, it updates the estimated MIDI parameters so that the output can be more similar to the input (e.g., if the pitch of the output is higher than the input, MIDI pitch bend control at its position is adjusted to compensate its difference). Our contribution is to use such an iterative process to imitate instrumental performances and explore dimensions of expressiveness that contribute to improve the naturalness of synthesized sounds.

InstListener consists of two parts: instrument performance analysis and synthesis, and performance comparison and micro-adjustment. The flow of the system is shown in Figure 1.

2.1. Feature Extraction

We first start with a feature extraction process that performs note onset detection as well as pitch extraction on audio signals. We perform note onset detection on the audio file based on convolution neural network proposed in [28] through the madmom python package [29]. We then use probabilistic YIN (pYIN) [30] algorithm through sonic annotator [31] toolkit for extracting note pitches and pitch contours because pYIN retains a smoothed pitch contour, preserving fine detailed melodic feature of instrumental performance. We then extract the energy component of the performance by computing the root-mean-square energy (RMSE) from the input audio file using the python package librosa [32].

2.2. Parameter mapping

Next, we map from acoustic features into discrete parameters for MIDI. We map the pitch contour into MIDI message. Unlike the piano, a pitched monophonic instrument (such as a saxophone or a clarinet) has continuous pitch contours rather than discrete ones. Thus, to reproduce nature expressive performance, we utilize the pitch bend control in MIDI file to reproduce a complete pitch contour. Given a pitch contour and a series of note onset times, we average the pitch for each note within the note duration distinguished by note onsets, to be further converted into MIDI note number. Based on the pitch contour information, we calculate the deviation of the actual pitch at certain time through the note from the MIDI note number, to be encoded as pitch bend information in the MIDI. Then, we map the RMSE into MIDI velocity level through linear mapping (with the maximum value corresponding to 127 as initial settings). Finally, we convert all the above information into the output MIDI file using pretty_midi python package[1].

2.3. Iterative listening process

Once a MIDI file imitating the original recording is generated, InstListener synthesizes it to generate an audio file. It uses pyFluidSynth[2] with soundfonts as a MIDI synthesizer in our current implementation. It then analyzes the MIDI synthesized audio file to obtain its acoustical features, which are then compared with acoustical features of the original input audio file to update the parameters to make the output more similar to the input. This iterative updating process is repeated until the parameters could converge. Or, we could stop repeating after a fixed number of iterations.

We use the pitch contour as one of the main acoustical features in the iterative process for comparison. During the comparison, we perform dynamic time warping (DTW) [33] between the pitch contour of the input and the pitch contour of the output. By using the DTW, InstListener adjusts and updates not only pitch contours, but also onset times because musical notes (and their onset times and durations) should be temporally moved in order to adjust the pitch alignment. In this way, the iterative process could contribute to improve the accuracy of musical note detection.

For this DTW, we want to find a mapping path

$$\{(p_1, q_1), (p_2, q_2), ..., (p_k, q_k)\} \tag{1}$$

such that the distance on this mapping path

$$\sum_{i=1}^{k} |t(p_i) - r(q_i)| \tag{2}$$

is minimized, with certain constraints as indicated in the DTW algorithm. As illustrated in Figure 2, the pitch contour got adjusted to approximate the original pitch contour through the iterations. InstListener automatically adjusts the onset time along with pitch information by minimizing such distance.

We also use the RMSE as an acoustical feature in the iterative process for comparison. We perform the same iterative and comparison process to adjust the MIDI velocity and MIDI volume control by minimizing the mean square error between the RMSE of the

[1]https://github.com/craffel/pretty-midi

[2]https://pypi.python.org/pypi/pyFluidSynth

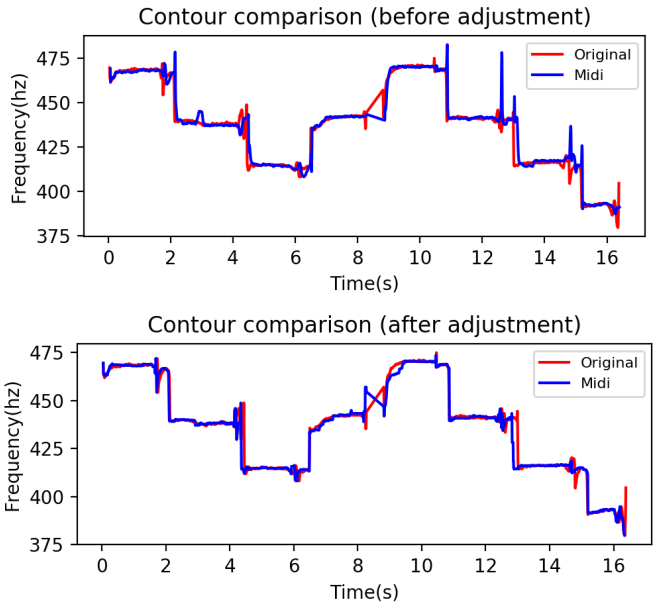

Fig. 2: Pitch contour and onset information. Top: before iterative adjustments. Bottom: after InstListener's iterative process (two contours get closer).

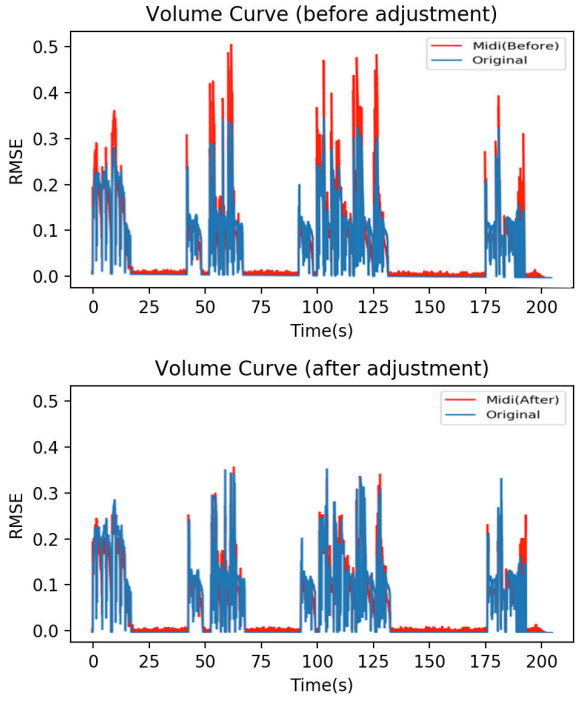

Fig. 3: Volume curve. Top: before iterative adjustments. Bottom: after InstListener's iterative process (two contours get closer).

original input performance and the RMSE of the MIDI-synthesized performance through least-square fitting. Figure 3 illustrates this adjustment of the MIDI velocities and volume control.

3. EXPERIMENTS

Musical expression is not yet an explicit action to measure. We propose our own method of evaluation. To evaluate different parameters and conditions, we implemented and conducted our experiments using a crowdsourcing platform, Amazon Mechanical Turk (MTurk)[3].

We first evaluated the similarity between the original input performance and each MIDI performance (MIDI rendition) synthesized by InstListener in order to compare different methods and conditions. We then evaluated the naturalness of the musical performances synthesized by InstListener for a further perceptual evaluation. We asked the turkers[4] to compare different renditions of expressive synthesized musical performances and the original input recording, and to rate how close (similar) each synthesized performance is to the original performance, as well as how natural do they think each performance is.

To avoid unreliable random behaviors that could happen on crowdsourcing tasks, we applied a pre-screening process by using a listening test to validate their normal hearing condition and behaviors. We used the following criteria to discard undesirable results from unreliable turkers:

1. We discarded turkers who did not pass the listening test. Our listening test consisted of three audio segments, each consisting of several sine tones. Each turker was asked to report the number of tones in each segment. We discarded the results from turkers who did not report them correctly.
2. We discarded turkers who finished the task much faster than the total time of the musical performances.

3.1. Experiment 1: Similarity perception test

In this similarity test experiment, the turkers were asked to listen to eleven sets of musical performances. For each set, they were asked to listen to the original input recording of a musical performance. Then given five different synthesized performances (renditions) imitating the same input, they were asked to rate how similar the current rendition is compared with the original performance in terms of musical expression. In our instruction, we described that, "by musical expressions, we refer to features such as musical dynamics, pitch contours, or overall musical gesture feelings." They were to rate the similarity on a scale of 1 to 7, with 7 meaning almost the same as the original performance, while 1 means very different from the performance in terms of musical expressions.

The five different renditions of the same original performance include: (1) DeadPan, MIDI without micro time adjustment of note onset and dynamic level as indicated by performers, (2) MIDI with velocity and without pitch bend information, (3) MIDI with velocity and pitch bend information, (4) InstListener with an expressive musical performance rendition with velocity and pitch bend imitating the original performance, and (5) Original input performance played be musicians.

We recruited a total of 50 turkers in the similarity listening test through MTurk. Each turker was paid for an amount of $0.5 for

[3]https://requester.mturk.com/

[4]We use a term "turker" to refer to a subject (crowdsourcing worker) that did our listening experiment on MTurk.

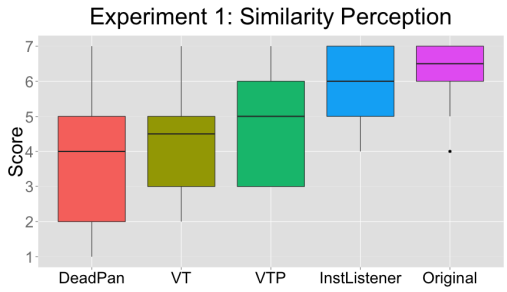

Fig. 4: Box Plot of Similarity Measurement test. DeadPan: MIDI without dynamics and that was quantized to $1/8$ note. VT: MIDI that incorporates velocity and timing information. VTP: Adding pitch bend information in addition to velocity and timing. InstListener: MIDI rendition after the iterative process. Original: recording from the original input performances by musicians.

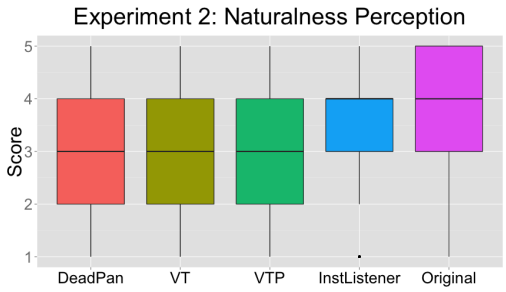

Fig. 5: Box Plot of Expressiveness and Naturalness Perceptual test. DeadPan: MIDI without dynamics and that was quantized to $1/8$ note. VT: MIDI that incorporates velocity and timing information. VTP: Adding pitch bend information in addition to velocity and timing. InstListener: MIDI rendition after the iterative process. Original: recording from the original input performances by musicians.

completing the task. Each task lasted for 20 to 30 minutes. In addition to the pre-screening process, we further excluded results from turkers who rated the original performance under the score of 5 out of 7 because we think they were unable to distinguish musical expressions for the purpose of our paper. The general pre-screening filtered out 4 who did not report the number of tones correctly, and 3 who completed too fast, and this task-specific pre-screening filtered out 22 out of 50 turkers. We thus included a total of 31 turkers into our experiment.

The result is shown in Figure 4. By filtering out unreliable turkers, we found that the original musical performances were scored the highest and DeadPan MIDI renditions were scored the lowest, as we expected. While adding the velocity and timing information contributes to the higher similarity of musical expressions, adding the degree of micro-tuning pitch contour reduces the variation of perception among the turkers. Finally, after the iterative parameter estimation process, InstListener was scored the highest as the most similar to the original recording in terms of musical expressions.

3.2. Experiment 2: Naturalness Perception Test

We are further interested in features that contribute to natural and expressive musical performances as perceived by human. In this experiment, we asked another batch of turkers to listen to the same eleven sets of performances. For each set, they were asked to rate the naturalness of a musical performance. The experiment lasted 20 to 30 minutes, and the turkers were paid for an amount of $1.5. They were asked to rate the naturalness on a scale of 1 to 5, with 5 meaning that the performance is very natural and expressive, while 1 means the performance sounds like robotic performances. We used a scale of 5 instead 7 because we think that the naturalness and expressiveness are too hard to be rated using many scales.

In this experiment, we collected responses from a total of 50 turkers. Two of them were discarded because they failed the listening test and were not qualified to be included.

As we can see from Figure 5, the non-expressive deadpan MIDI rendition was scored the lowest (very robotic) by the turkers. While the original performance was scored the highest, as we gradually added parameters to the MIDI rendition, we were able to see an perceptual improvement. When the velocity and pitch bend information were added, the scores became higher. Furthermore, InstListener with the iterative process was scored higher, though not as comparable as the original one.

The result is shown in Figure 5. We found that adding the velocity, timing, and pitch information to the deadpan did not contribute to the naturalness perception. The score for InstListener, however, still got the best score among the others except for the original performance. We thus confirmed that the naturalness of the synthesized performances by InstListener was not low and was higher than other renditions without the iterative process.

4. DISCUSSION AND CONCLUSION

We present InstListener, a system that converts expressive musical performances into the MIDI-based parametric representation. InstListener has a potential to enable people to easily transfer musical expressions onto other musical instruments by only changing the timbre space (e.g., MIDI program number). In addition to rendering with such a variety of timbres, people can also intentionally change some portions of the estimated parameters to achieve a different musical style (e.g., keep the same velocity while changing the pitch contour or timbre separately). In this way, the contributions of this paper are not only to imitate, parameterize, and aggregate musical expressions by human performers, but also to control musical expressions more flexibly to achieve and explore various expressions.

We evaluated our system from the perceptual point of view. We first evaluated the success of imitating and approximating the original performance not only at the note level, but also in terms of musical expressions. Our experimental results showed that InstListener imitated the original musician's performance well, and the results got much improved after our iteration process. However, even if a synthesized performance is similar enough to its original performance, the naturalness of the synthesized performance is not necessarily high. We therefore explored the naturalness of the estimated parameters through the MIDI rendition, and confirmed that the synthesized performance imitated by InstListener was natural enough. Future work includes constructing performers' models using parameterized controls, exploring how humans express musical expressions and features that contribute to expressive performances.

5. ACKNOWLEDGEMENT

This work was supported in part by JST ACCEL Grant Number JPMJAC1602, Japan.

6. REFERENCES

[1] Kate Hevner, "Experimental studies of the elements of expression in music," *The American Journal of Psychology*, vol. 48, no. 2, pp. 246–268, 1936.

[2] Neil Todd, "A model of expressive timing in tonal music," *Music Perception: An Interdisciplinary Journal*, vol. 3, no. 1, pp. 33–57, 1985.

[3] Caroline Palmer, "Anatomy of a performance: Sources of musical expression," *Music perception: An interdisciplinary journal*, vol. 13, no. 3, pp. 433–453, 1996.

[4] Gerhard Widmer and Werner Goebl, "Computational models of expressive music performance: The state of the art," *Journal of New Music Research*, vol. 33, no. 3, pp. 203–216, 2004.

[5] Alexis Kirke and Eduardo Reck Miranda, "A survey of computer systems for expressive music performance," *ACM Computing Surveys (CSUR)*, vol. 42, no. 1, pp. 3, 2009.

[6] Joel Chadabe, "Electric sound: The past and promise of electronic music," 1997.

[7] Max V Mathews, "The digital computer as a musical instrument," *Science*, vol. 142, no. 3592, pp. 553–557, 1963.

[8] Arthur A Reblitz, *Player Piano: Servicing and Rebuilding*, Vestal Press, 1997.

[9] Patrik N Juslin, "Five facets of musical expression: A psychologist's perspective on music performance," *Psychology of Music*, vol. 31, no. 3, pp. 273–302, 2003.

[10] Giovanni De Poli, "Methodologies for expressiveness modelling of and for music performance," *Journal of New Music Research*, vol. 33, no. 3, pp. 189–202, 2004.

[11] Caroline Palmer, "Music performance," *Annual review of psychology*, vol. 48, no. 1, pp. 115–138, 1997.

[12] Anders Friberg, Roberto Bresin, and Johan Sundberg, "Overview of the kth rule system for musical performance," *Advances in Cognitive Psychology*, vol. 2, no. 2-3, pp. 145–161, 2006.

[13] Guerino Mazzola, *Musical performance: A comprehensive approach: Theory, analytical tools, and case studies*, Springer Science & Business Media, 2010.

[14] Giovanni De Poli, Antonio Rodà, and Alvise Vidolin, "Note-by-note analysis of the influence of expressive intentions and musical structure in violin performance," *Journal of New Music Research*, vol. 27, no. 3, pp. 293–321, 1998.

[15] Jeffrey C Smith, "Correlation analyses of encoded music performance," 2013.

[16] Kenta Okumura, Shinji Sako, and Tadashi Kitamura, "Stochastic modeling of a musical performance with expressive representations from the musical score.," in *ISMIR*, 2011, pp. 531–536.

[17] Roger B Dannenberg, Hank Pellerin, and Itsvan Derenyi, "A study of trumpet envelopes," 1998.

[18] Istvan Derenyi and Roger B Dannenberg, "Synthesizing trumpet performances," *Computer Science Department*, p. 500, 1998.

[19] Alf Gabrielsson, "Interplay between analysis and synthesis in studies of music performance and music experience," *Music Perception: An Interdisciplinary Journal*, vol. 3, no. 1, pp. 59–86, 1985.

[20] Sergio Canazza, Giovanni De Poli, Carlo Drioli, Antonio Rodà, and Alvise Vidolin, "Audio morphing different expressive intentions for multimedia systems," *IEEE Multimedia*, , no. 3, pp. 79–83, 2000.

[21] Rumi Hiraga, Roberto Bresin, Keiji Hirata, and Haruhiro Katayose, "Rencon 2004: Turing test for musical expression," in *Proceedings of the 2004 conference on New interfaces for musical expression*. National University of Singapore, 2004, pp. 120–123.

[22] Sofia Dahl and Anders Friberg, "Visual perception of expressiveness in musicians' body movements," *Music Perception: An Interdisciplinary Journal*, vol. 24, no. 5, pp. 433–454, 2007.

[23] Eduardo R Miranda, Alexis Kirke, and Qijun Zhang, "Artificial evolution of expressive performance of music: an imitative multi-agent systems approach," *Computer Music Journal*, vol. 34, no. 1, pp. 80–96, 2010.

[24] Jessika Karlsson and Patrik N Juslin, "Musical expression: An observational study of instrumental teaching," *Psychology of Music*, vol. 36, no. 3, pp. 309–334, 2008.

[25] Patrik N Juslin and Petri Laukka, "Expression, perception, and induction of musical emotions: A review and a questionnaire study of everyday listening," *Journal of New Music Research*, vol. 33, no. 3, pp. 217–238, 2004.

[26] Tomoyasu Nakano and Masataka Goto, "VocaListener: A singing-to-singing synthesis system based on iterative parameter estimation," *Proc. SMC*, pp. 343–348, 2009.

[27] Tomoyasu Nakano and Masataka Goto, "VocaListener2: A singing synthesis system able to mimic a user's singing in terms of voice timbre changes as well as pitch and dynamics," in *IEEE ICASSP*, 2011, pp. 453–456.

[28] Jan Schluter and Sebastian Bock, "Improved musical onset detection with convolutional neural networks," in *IEEE ICASSP*, 2014, pp. 6979–6983.

[29] Sebastian Böck, Filip Korzeniowski, Jan Schlüter, Florian Krebs, and Gerhard Widmer, "madmom: a new python audio and music signal processing library," in *Proceedings of the 2016 ACM on Multimedia Conference*, 2016, pp. 1174–1178.

[30] Justin Salamon and Emilia Gómez, "Melody extraction from polyphonic music signals using pitch contour characteristics," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 6, pp. 1759–1770, 2012.

[31] Chris Cannam, Michael O. Jewell, Christophe Rhodes, Mark Sandler, and Mark d'Inverno, "Linked data and you: Bringing music research software into the semantic web," *Journal of New Music Research*, vol. 39, no. 4, pp. 313–325, 2010.

[32] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings in the 14th python in science conference*, 2015, pp. 18–25.

[33] Donald J Berndt and James Clifford, "Using dynamic time warping to find patterns in time series.," in *KDD workshop*. Seattle, WA, 1994, vol. 10, pp. 359–370.