QUERY-BY-CONDUCTING: AN INTERFACE TO RETRIEVE CLASSICAL-MUSIC INTERPRETATIONS BY REAL-TIME TEMPO INPUT

Akira Maezawa,[†] Masataka Goto[‡] and Hiroshi G. Okuno[†]

[†]Dept. of Intelligence Science and Technology
Graduate School of Informatics, Kyoto University
Sakyo-ku, Kyoto 606-8501 Japan
{amaezaw1, okuno}@kuis.kyoto-u.ac.jp

[‡]National Institute of Advanced Industrial
Science and Technology (AIST)
Tsukuba, Ibaraki 305-8568 Japan
m.goto@aist.go.jp

ABSTRACT

This paper presents an interface for finding *interpretations* of a user-specified music, *Query-by-Conducting*. In classical music, there are many interpretations to a particular piece, and finding "the" interpretation that matches the listener's taste allows a listener to further enjoy the piece. The critical issue in finding such an interpretation is the way or interface to allow the listener to listen through different interpretations. Our interface allows a user, by swinging a conducting hardware interface, to conduct the desired global tempo along the playback of a piece, at any time in the piece. The real-time conducting input by the user dynamically switches the interpretation being played back to the one closest to how the user is currently conducting. At the end of the piece, our interface ranks each interpretation according to how close the tempo of each interpretation was to the user input.

At the core of our interface is an automated tempo estimation method based on audio-score alignment. We improve tempo estimation by requiring the audio-score alignment of different interpretations to be consistent with each other. We evaluate the tempo estimation method using a solo, chamber, and orchestral repertoire. The proposed tempo estimation decreases the error by as much as 0.94 times the original error.

1. INTRODUCTION

Classical music is unique in that many audio recordings exist for a given piece of music. For example, as of March 2010, a search on an on-line shopping site for "Mendelssohn Violin Concerto" returns 1200+ hits, or that of "Beethoven Spring Sonata" returns 300+ hits. Each of these recordings is an acoustic rendition of a particular music score, embodied by an unique interpretation of the performer. Finding an interpretation that matches the listener's taste is an important aspect of enjoying classical music. However, searching for such recording is tiresome because it requires the listener to listen through the same piece many times.

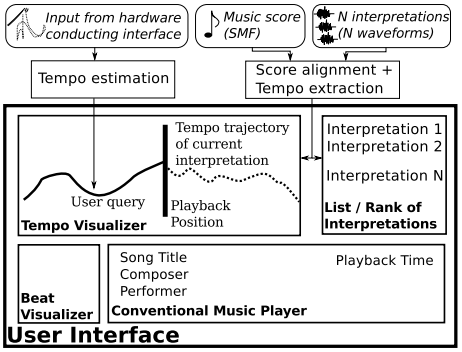


Figure 1: System diagram of *Query-by-Conducting*.

Our goal is to retrieve *interpretations*[1] on the basis of various aspects of music interpretation, similar to content-based music information retrieval (CBMIR), which retrieves *pieces* on the basis of various aspects of music such as rhythm and timbre [2, 3]. What constitutes musical interpretation is a difficult question, though it seems that musicians express interpretation by manipulating the tempo, the timbre, or bringing out interesting melodic lines. This paper focuses on the global tempo – the average tempo of a piece over a few beats. We believe tempo is an aspect of music interpretation that many listeners take note of. Studies in music cognition suggest that similarity of interpretations is strongly reflected in global tempo [4], and many studies are motivated by the significance of tempo on interpretation [1, 5].

We present *Query-by-Conducting*, a new interface for finding *interpretations* of a user-specified music by conducting the global tempo. The interface reads, as the music score, a standard MIDI file of a piece of music, and different interpretations of the music score as audio files. The interface facilitates playback, visualization and query of interpretations by supporting the following features, as shown in Figure 1:

1. Visualization of global tempi of the interpretations
2. Hardware conducting interface (a *Nintendo Wii* remote) for intuitively entering, along the playback of a piece, the user's tempo query in real time
3. Ranking and retrieval of interpretations on the basis of the similarity between each interpretation and the current tempo query entered by the user.

Visualization allows the user to view the range of interpretations available, and thus, the valid range of tempo

[1] We shall use the term "interpretation" to mean a rendition of a particular symbolic representation of music, as per [1].

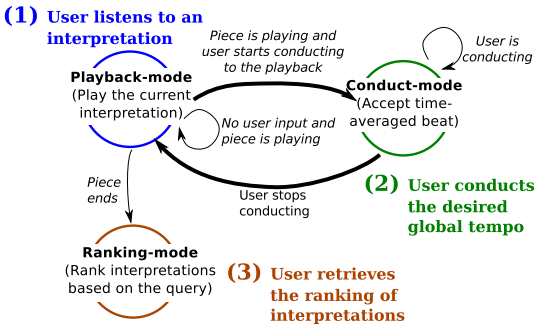


Figure 2: System-level state diagram. The interface alternates between (1) and (2) until the piece ends.

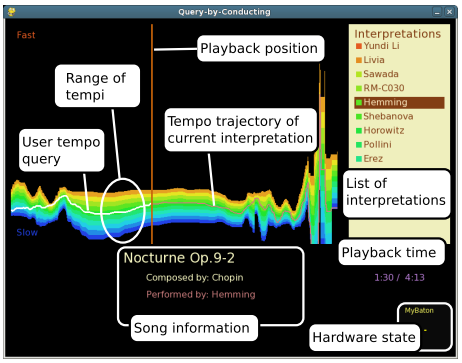


Figure 3: The interface in **playback-mode** visualizes the tempo along playback of an interpretation.

in which the user should conduct to obtain a meaningful query. As a particular interpretation is played back, the user may become dissatisfied with its tempo. Then, the user would "conduct" the desired tempo using the hardware conducting interface. Based on the user's tempo input, the interface retrieves and switches the interpretation being played back to one that is closest to the input. The user may stop conducting if the current interpretation is satisfactory. At the end of playback, the system ranks each interpretation on the basis of how similar the overall global tempo trajectory was to the user's conducting.

Our interface differs from existing conducting interfaces [6–9] in three respects. First, we use conducting device to switch the interpretation being played back, instead of specifying the tempo of the entire piece. The user has the freedom of either listening to a piece, or conducting the tempo of an interpretation that the user wants to listen. Second, our method allows the user to control only the global tempo instead of local tempo or dynamics. We believe that such restriction is an effective way to retrieve a particular interpretation; global tempo is easy for a typical user to specify, but specifying local tempo requires a precise control of the tempo. The notion of restricting the user control to a few dimensions has been proposed in other studies aimed at easily manipulating expressive music [10]. Finally, our conducting interface is meant to retrieve a particular interpretation to play back, whereas most conducting interfaces are aimed at real-time temporal manipulation of a particular audio signal. Unlike query by tapping [11], which uses *rhythm pattern* as the query, as our method uses the *tempo* as the query.

The interface relies on tempo estimation that is determined through audio-score alignment. In existing studies [1, 12–14], audio-score alignment was created using only the information obtained from the audio of interest and the score. There may be errors in the alignment, but given one alignment, there is no way of knowing where an error is. When aligning multiple interpretations, however, it is also possible to create audio-score alignment by aligning the score to some *other* audio, and then aligning that audio to the audio of interest. Thus, given N interpretations to align, N unique audio-score alignments to *one* interpretation can be generated. We use these multiple audio-score alignments generated to estimate the true audio-score alignment that is error-free.

A video demonstration of our interface is available at http://www.youtube.com/QueryByConducting

2. INTERFACE DESIGN

The interface offers functions in a conventional music playback interface such as playback and rewind. Moreover, it features a visualizer of global tempi of various interpretations, a conducting hardware to enter the global tempo query in real time, and retrieval of interpretations on the basis of the user query. As shown in the state diagram in Figure 2, our system alternates between playing back the current interpretation ("playback-mode"), and accepting user conducting and retrieving appropriate interpretation to play back ("conduct-mode"). At the end, our system ranks each interpretation on the basis of the tempo ("ranking-mode").

Figure 3 shows the interface during playback (**playback-mode**). Bottom of the screen displays the title, the performer and the playback time, similar to conventional music playback interface. Top of the screen visualizes the global tempi, and presents each interpretation sorted in descending order of the current global tempo. Bottom right shows the state of the conducting interface.

As the piece is played back, the user may become dissatisfied with the tempo of the piece ("I liked the introduction, but the development section is too slow," a user might think). As shown in Figure 4, the interface allows a user to "conduct" the desired global tempo in **conduct-mode**, in real time. In **conduct-mode**, the interface accepts beat input from the conducting hardware interface, and also visualizes the beat at the bottom left to facilitate proper conducting. The entered tempo is used as a query to retrieve the interpretation whose global tempo is closest to what the user conducts, and to switch the current playback to it. This mode offers the user an active listening experience by constantly retrieving and cross-fading the playback to interpretation that plays like how the user is conducting.

At the end of the piece, the interface enters the **ranking-mode**, and ranks each interpretation based on how similar each interpretation was to the user's overall conducting. The ranking is presented to the user.

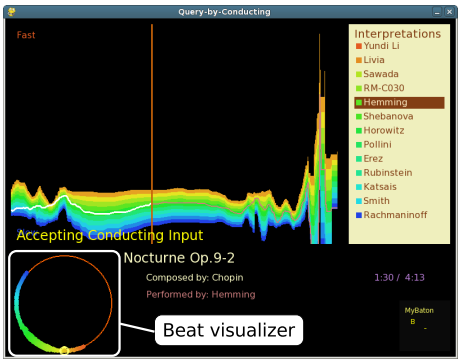


Figure 4: The interface in **conduct-mode** accepts user's conducting using a controller, and switches interpretations being played back. Beat visualizer facilitates user's conducting.

2.1 Interpretation Visualizer

Top half of the interface (in Figure 3) is the interpretation visualizer. It shows the tempi of different interpretations, along with tempo trajectory of the interpretation that is being played back and the user query.

Figure 5 shows the visualizer in further detail. It presents tempo information against time. To allow the user to view the detailed tempo near the current playback position as well as the tempo of the entire piece, we distort the normalized x-coordinate, $x(t)$. $x(t)$ is distorted such that the vicinity of current playback position t_c is zoomed like a lens. Let t, the current beat of playback, be defined for $[0, t_l]$, where t_l is the duration of the piece, and the range of $x \in [0, 1]$. Then, the visualizer applies the following function:

$$x(t) = \frac{t}{2t_l} + \frac{1}{2}\frac{\exp\left(\frac{t-t_c}{T}\right)}{\exp\left(\frac{t-t_c}{T}\right)+1} \quad (1)$$

This shows approximately T-neighborhood of the current playback position in more detail than the rest. T is chosen to be four quarter notes.

A vertical straight line indicates the current playback position. Line segment to the left of the current playback position is the past tempo trajectory of the user's global tempo query. Line segment to the right of the current playback position is the future tempo trajectory of the interpretation that is being played back. This way, the user is able to view the query entered so far, and how the current interpretation will unfold.

To show the range of possible interpretations, the range of global tempi is expressed as a colorful strip, overlayed to to the line segments described above. The strip is colored using a gradation of hue angle, such that fast tempo is associated with small hue (orange), and slow tempo with large hue (blue). At beat t, given the slowest tempo $\tau_{\min}(t)$, fastest tempo $\tau_{\max}(t)$, and some tempo in between, $\tau(t)$, we set the hue to the following angle:

$$\text{hue}(t) = 240^\circ - \frac{\tau(t) - \tau_{\min}(t)}{\tau_{\max}(t) - \tau_{\min}(t)} 230^\circ \quad (2)$$

Right half of the visualizer prints the performer of each interpretation, sorted in descending order of the current global tempo. The interpretation that is being played back is highlighted. Next to each name, a box whose hue value is as described in Equation (2) is painted.

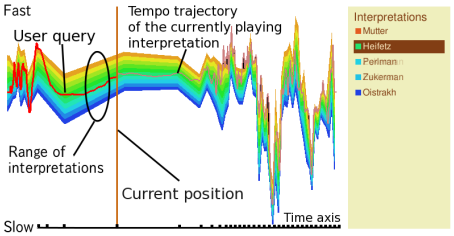


Figure 5: Interpretation visualizer shows the tempo trajectory of the current interpretation, how the user has conducted, and the range of tempi.

2.2 Hardware Conducting Interface

The hardware conducting interface detects beat from an accelerometer embedded in the hardware controller, and converts it into tempo. The user interface shows beat visualizer to facilitate tempo entry.

2.2.1 Beat detection

We accept the user's conducting query using a game controller that features a 3-axis accelerometer (a Nintendo Wii controller). Our system detects beat by checking for peaks in the axis vertical to the controller. Such peak is generated when the controller is flicked up, as a conductor would flick the baton to indicate the beat.

Once a beat is detected, the accelerometer input is ignored for 200msec to prevent false triggering. Therefore, our system accepts tempo of up to 300 beats-per-minute (BPM), which is sufficient for virtually all classical music.

2.2.2 Converting beat input to tempo query

To specify a new tempo, the user must conduct a tempo different from the playback. We observed that people tends to conduct *not* in the desired tempo, but instead ahead or behind of the beat of the playback to indicate faster or slower tempo relative to the current playback. We conjectured that such phenomenon occurs because people are distracted by the downbeat of the playback, and sets the desired beat location *relative* to the last downbeat he/she has heard. Therefore, we convert the offset of the user's conducting with respect to the beat of the playback, to the desired tempo in BPM. Suppose the user conducts Δ_t behind the beat. Then, supposing the current BPM of the playback is BPM_0, we convert Δ_t to user-specified tempo, BPM, as follows:

$$BPM = BPM_0 \frac{1}{\frac{\Delta_t}{60/BPM_0} + 1} \quad (3)$$

The average of user-specified BPM over four beats is used as the query.

2.2.3 Beat visualizer

We observed that, in a preliminary experiment using a few test subjects, people did not always have a clear sense of rhythm, and had trouble finding where the beat is. This was especially true for music whose instrumentation did not include instrument with strong attack and decay, such as the piano or plucked strings.

To facilitate tempo input, we display a beat visualizer, as shown at the bottom left of Figure 3, and in detail in

Figure 6: Visualizer for facilitating tempo input. The color bar rotates in synchrony with the beat so that the user could easily grasp the downbeat.

Figure 6. The visualizer has a colored stripe that rotates around the large circle, in synchrony with the beat of the playback. At each downbeat, the stripe crosses the small circle at the bottom. The arc-length of the rotating stripe corresponds to the range of global tempi at the current beat, and the hue is calculated using Equation (2). Therefore, if the user wants to switch to fast interpretation, for example, from tempo shown as green in the tempo visualizer to orange-colored tempo, the user could flick the controller as the orange-colored segment of the arc crosses the small circle at the bottom.

2.3 Interpretation Retriever

After the user has finished listening through a piece, the interface ranks, in **ranking-mode**, each interpretation on the basis of the similarity between the tempo trajectory of the interpretation and the user query.

We use the tempo trajectories of the interpretations that were played as the query. For example, if the user listened to interpretation x for the first minute and y for the next two minutes, our query would consist of the tempo trajectory of interpretation x for the first minute and y for the next two.

Let us define the dissimilarity score of the user query and each interpretation. Let $\tau_i(t)$ be the global tempo trajectory of the ith interpretation, and $\tau_q(t)$ be the query tempo trajectory. Then, we define tempo dissimilarity for interpretation i, r_i as follows:

$$r_i = \frac{1}{T}\int_0^T \left(\frac{\tau_i(t) - \tau_q(t)}{\tau_q(t)}\right)^2 dt \tag{4}$$

The interpretations are sorted in the ascending order of tempo dissimilarity, as shown in Figure 7. A transparency value is associated to each interpretation being drawn, such that interpretation with lowest dissimilarity is opaque, and the highest transparent. Moreover, dissimilarity measure that is inverted, shifted and scaled between 0 and 1 is shown next to each interpretation.

3. TEMPO EXTRACTION METHOD

Global tempo extraction is based on evaluating the audio-score alignment. Since accurate alignment is essential for accurate tempo estimation, we propose a method to improve the audio-score alignment.

3.1 Initial Audio-Score Alignment

The initial audio-score alignment is based on dynamic time-warping (DTW) using chroma vector as the feature, similar to other works [13–15].

Let $c_k^{(t)}$ be a 12 dimensional vector that contains the chroma vector computed for tth audio frame of the kth

Figure 7: **Ranking-mode** ranks each interpretation based on the tempo similarity, presents similar interpretations as opaque and dissimilar ones transparent.

interpretation. Let $c_S^{(t)}$ be a 12 dimensional chroma vector computed from the music score at tick t. We generate the alignment from the score to the kth interpretation, denoted $M_{k\leftarrow s}$, or from interpretation i to j, denoted $M_{j\leftarrow i}$. To generate $M_{k\leftarrow s}$, a similarity matrix $R_{k\leftarrow s}$ is first computed. Let N_s be the number of ticks in the music score and N_k be the number of audio frames contained in interpretation k. Let $R_{k\leftarrow s}$ be a N_s-by-N_k matrix, whose element i, j contains:

$$R_{k\leftarrow s}(i,j) = 1 - \frac{c_s^{(i)} \cdot c_k^{(j)}}{\|c_s^{(i)}\| \cdot \|c_k^{(j)}\|} \tag{5}$$

Next, we find the alignment path using DTW. Formally, we define a cost matrix N_s-by-N_k matrix, as follows:

$$C_{k\leftarrow s}(i,j) = R_{k\leftarrow s}(i,j) + \min \begin{cases} C_{k\leftarrow s}(i-1,j) \\ C_{k\leftarrow s}(i,j-1) \\ C_{k\leftarrow s}(i-1,j-1) \end{cases} \tag{6}$$

where for all t, $C_{k\leftarrow s}(t,-1) = C_{k\leftarrow s}(-1,t) = 0$. Next, we determine the parametric representation of the audio-score alignment, $M_{k\leftarrow s}^{(t)}$, by backtracking the cost matrix. First, we set $M_{k\leftarrow s}^{(0)}$ to (N_s, N_k), and update in the following manner while incrementing t, until $M_{k\leftarrow s}^{(t)} = (0,0)$:

$$\begin{aligned} M_{k\leftarrow s}^{(t+1)} &:= \operatorname*{argmin}_{(I,J)\in S} C_{k\leftarrow s}(I,J) \\ S &= \{(i-1,j),(i,j-1),(i-1,j-1)\} \end{aligned} \tag{7}$$

where $(i,j) = M_{k\leftarrow s}^{(t)}$. Audio-audio alignment from interpretation i to j, $M_{j\leftarrow i}$, can be achieved in the same way, by computing the similarity matrix between chroma vector sequence of interpretation i and j.

3.2 Improving Audio-Score Alignment

We improve audio-score alignment by requiring the alignments of different interpretations to be consistent with each other. Given one music score and N interpretations, there are N possible paths to generate the alignment from the music score to interpretation i, as shown in Figure 8. Namely, in addition to the direct mapping from the score to interpretation i, it is also possible to generate mapping from the score to interpretation j (audio-score alignment), which is then mapped by using the map from interpretation j to i (audio-audio alignment). Ideally, all N paths from the score to an interpretation should be identical. In reality, however, they are not because they are generated using different similarity matrices.

In order to generate a map from the score to some interpretation i via interpretation j, $M_{i\leftarrow j} \circ M_{j\leftarrow S}$, both $M_{j\leftarrow S}$ and $M_{i\leftarrow j}$ must be one-to-one, but the alignments generated in the previous section are not.

Therefore, we trace, over the alignment determined in the previous section, a new map that is one-to-one. We perform the following procedure for each alignment between some interpretation (or score) s and k:

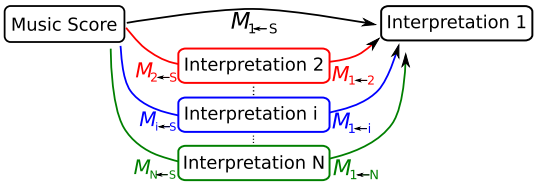


Figure 8: By combining two alignments, there are multiple ways to align the score to an interpretation.

1. Set $t = 0$, and the initial point of the refined alignment $\tilde{M}^{(t)}_{k\leftarrow s}$ to $(0, 0)$.
2. For $\epsilon < \theta < \frac{\pi}{2} - \epsilon$, compute the following cost function:
$$c(\theta) = E_{q\sim \exp(-3q/Q)}[\min_n d^{(t)}_n(q, \theta)] \quad (8)$$
where
$$d^{(t)}_n(q, \theta) = \|\tilde{M}^{(t)}_{k\leftarrow s} + q\,(\cos(\theta), \sin(\theta)) - M^{(n)}_{k\leftarrow s}\| \quad (9)$$
Q is chosen to be 20 frames, and ϵ to be $\pi/20$ radian. θ is evaluated every $\pi/20$ radians.
3. Update $\tilde{M}_{k\leftarrow s}$ as follows, for some $\Delta r \in (0, 1]$:
$$\tilde{M}^{(t+1)}_{k\leftarrow s} := \tilde{M}^{(t)}_{k\leftarrow s} + \Delta r\left(\cos(\hat{\theta}), \sin(\hat{\theta})\right) \quad (10)$$
$$\hat{\theta} = \operatorname*{argmin}_\theta c(\theta)$$
We chose $\Delta r = 1$ frame.
4. Exit if $\tilde{M}^{(t)}_{k\leftarrow s} \cdot (1, 0) \geq N_s$ or $\tilde{M}^{(t)}_{k\leftarrow s} \cdot (0, 1) \geq N_k$.
5. Set $t := t + 1$, and go to 2.

We assume that observed alignments are corrupted by independent and identically distributed noise that follows the Laplace distribution with location parameter $\hat{M}_{i\leftarrow S}(t)$ and scale parameter b, for each beat t:
$$p(t) = \exp\left(-\|M_{i\leftarrow S}(t) - \hat{M}_{i\leftarrow S}(t)\|/b\right)/2b \quad (11)$$
and likewise for $M_{i\leftarrow j} \circ M_{j\leftarrow S}$ for $j \neq i$. We interpret $\hat{M}_{i\leftarrow S}$ as the underlying "correct" alignment that generates $M_{i\leftarrow S}$ and $M_{i\leftarrow j} \circ M_{j\leftarrow S}$. Since an estimator of $\hat{M}_{i\leftarrow S}$ is the sample median, we update $M_{i\leftarrow S}$ as follows:
$$M_{i\leftarrow S}(t) := \text{median}(\{M_{i\leftarrow j} \circ M_{j\leftarrow S}(t)\}) \quad (12)$$
As will be shown in the experiment, iterating this step yields in improved alignment accuracy.

3.3 Tempo Extraction

The tempo is estimated by determining the slope of the audio-score alignment. We compute the tempo at MIDI tick t using alignment information obtained between tick $t - T$ to $t + T$ for $T > 0$. Only information at note onsets are used, as alignment results between two note onsets are not reliable. We choose T dynamically such that at least 20 audio frames that correspond to note onsets are within this range. Let $(s(p), a(p))$ contain a parametric representation of $M_{i\leftarrow S}$ that contain the audio frames chosen. s corresponds to the domain (tick of note onsets) and a the range (audio frame). Then, we compute the BPM at tick t, $\tau(t)$ by first finding the slope $m(t)$ of $(s(p), a(p))$ using linear regression, and multiplying its inverse by a scalar factor:
$$\tau(t) = \frac{1}{m(t)} \frac{\text{audio frame-per-minute}}{\text{ticks-per-beat}} \quad (13)$$

4. EXPERIMENTS

We evaluate the tempo estimation method, and retrieval of interpretation on the basis of global tempo query. We analyzed nine classical pieces of varying instrumentation. Of nine pieces, three are orchestral (denoted *orch-1* to *orch-3*), three are written for small ensemble (denoted *duo-1* to *duo-3*), and three are solo piano (denoted *solo-1* to *solo-3*). For each work, multiple interpretations (between four and thirteen) were obtained and their ground truth tempo data were entered using an in-house tempo entry utility.

Table 1: Average MSE (mean-squared error) improvement in thousandths (10^{-3}) after iterating Equation (12).

Piece (No. Interp.)	None	Iter. 1	Iter. 2	Iter. 10
solo-1 (13)	8.9	8.6	8.4	8.4
solo-2 (6)	17.3	15.0	13.0	12.7
solo-3 (5)	266.7	73.1	85.4	98.8
duo-1 (5)	4.5	3.9	3.7	3.8
duo-2 (4)	34.8	22.1	20.6	20.4
duo-3 (4)	185.4	12.5	10.2	10.2
orch-1 (5)	646.8	54.4	47.2	44.9
orch-2 (5)	231.5	14.7	13.3	13.2
orch-3 (5)	3941.6	1091.4	1038.3	833.2

4.1 Evaluation of Audio-Score Alignment

Let $\tau_g(t)$ be the ground truth tempo trajectory. Given an estimated tempo trajectory $\hat{\tau}(t)$, we evaluate the error using scaled mean squared error (MSE), defined as follows:
$$\text{MSE} = \frac{1}{T}\int_0^T \left(\frac{\tau_g(t) - \hat{\tau}(t)}{\tau_g(t)}\right)^2 dt \quad (14)$$
MSE can be considered as the dissimilarity measure between the ground truth and the estimated tempo.

Table 1 shows the average of MSE over all interpretation for each of the nine pieces, as the number of iterations of the update step (Equation (12)) is changed.

The results suggest that, first, our method is capable of decreasing the error, more so if the initial error is high. For example, *duo-3* has its error decreased by 0.94 times the original error, after ten iterations. Second, in most cases, iterating our method multiple times yields in decreased error. When the error increases with increased number of iterations, we believe that our assumption that alignments are corrupted by independent noise fails. For example, in pieces that involve unnotated *cadenza* (e.g. *solo-3*), incorrect alignment occurs consistently at the cadenza. Then, taking the median of such corrupted data yields not in the underlying "true" alignment, as our method posits, but some meaningless data instead.

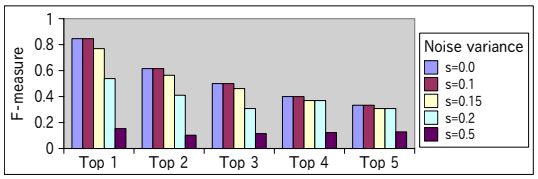
4.2 Evaluation of Music Query

We evaluate the robustness of our system against errors in conducting. When a user conducts like some interpretation i, the system should retrieve i as the most similar interpretation. Other results may be returned for two reasons:

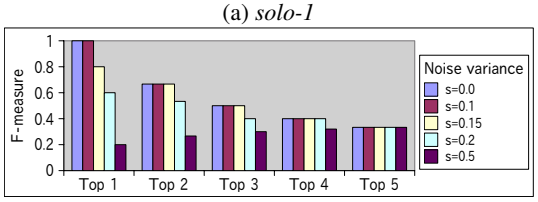
1. The user could not conduct the piece accurately enough to return the desired query.
2. Imprecision in tempo estimation method causes incorrect result to be returned.

In these cases, i may not be the most similar, but one of M most similar interpretations.

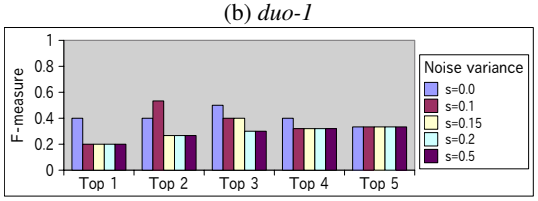
First, we synthesize an artificial query that models human errors in conducting, by adding a smooth noise to the ground truth tempo trajectory of each data. For each interpretation i, we use the following tempo trajectory as the query with some noise variance s :
$$\tau_{\text{query,i}}(t; s) = \tau_{g,i}(t) \cdot 2^{\sqrt{\frac{s}{L}}\sum_{l=0}^{L-1} n(t-l)} | L = 10 \quad (15)$$
$$n(t; s) \sim \mathcal{N}(0, 1)$$



(a) *solo-1*



(b) *duo-1*



(c) *orch-1*

Figure 9: Evaluation results when retrieving up to top 5 results that are similar to artificially generated query which deviates from the ground truth by variance s.

Next, we retrieve M interpretations that are most similar to the artificial query for each interpretation, and evaluate the performance of retrieval the F-measure. Let n be the number of interpretations correctly retrieved by the query. Let N be the total number of interpretations. Then, we let recall $R = n/N$, and precision $P = n/(N \times M)$. The F-measure F is $2PR/(P + R)$. We show the results from *solo-1*, *duo-1*, and *orch-1* in Figure 9 (a), (b), and (c).

Figure 9 (a) and (b) show that the system tolerates small error in conducting, of up to about $s = 0.15$, or 0.7 to 1.3 times the original tempo (three-sigma). Figure 9 (c), however, shows that the F-measure of *orch-1* is considerably lower than the other two.

Retrieving orchestral piece (*orch*'s) is difficult because there is very small variation in the two closest playing, and exacerbated by the particularly unreliable tempo estimation. We compute the smallest dissimilarity measure between ground truth tempo trajectories of any pair of interpretations. The smallest dissimilarity of *orch-1* is about 2×10^{-3}, *solo-1* is 20×10^{-3}, and *duo-1* is 23×10^{-3}. We similarly observed that for orchestral piece, the smallest dissimilarity is much smaller compared to that of chamber (*duo*'s) or solo (*solo*'s). On the other hand, we observe that the average MSE, as seen in Table 1, is substantially greater for orchestral pieces than chamber or solo.

These results suggest that our system retrieves the desired interpretation with robustness against minor errors in conducting, as long as the average MSE is small enough to differentiate the most similar pair of interpretations. The similarity of interpretation is typically influenced by the scale of orchestration, and the average MSE is influenced by the complexity of the ensemble, and the degree to which the interpretation deviates from the music score.

5. CONCLUSION

This paper presented *Query-by-Conducting*, an interface for finding *interpretations* of a given piece of music. It offers the listener an interactive experience of "conducting" the global tempo to dynamically tailor the interpretation played back to the user's choice. It moreover presents the listener with a ranking of interpretation based on how the user conducted through the piece, offering the listener with a list of interpretations whose tempi that the user might like, without the hassle of listening through various interpretations. The accuracy of tempo estimation method improved as a result of considering the consistency of audio-score alignment among different interpretations.

As future work, we would like to deal with aspects of music interpretation other than the global tempo, such as the local tempo deviation and emphasis of a particular melodic line. Integrating these aspects would further enhance the system's capability to retrieve the interpretation of choice. Furthermore, we would like to realize more ways to visualize and interact with various aspects of music interpretation, to allow a listener to further enjoy classical music.

Acknowledgment: This research was partially supported by Graint-in-Aid for Scientific Research (S) of the Ministry of Education, Culture, Sports, Scince and Technology (MEXT), and the CrestMuse Project of the Japan Science and Technology Agency (JST).

6. REFERENCES

[1] M. Müller *et al.* Towards automated extraction of tempo parameters from expressive music recordings. In *ISMIR '09*, pages 69–74, 2009.

[2] C.A.Michael *et al.* Content-based music information retrieval: Current directions and future challenges. volume 96, pages 668–696, April 2008.

[3] S. Dixon, F. Guyon and G. Widmer. Towards characterization of music via rhythmic patterns. In *ISMIR '04*, pages 509–516, 2004.

[4] R. Timmers. Predicting the similarity between expressive performances of music from measurements of tempo and dynamics. *JASA*, 117(1):391–399, 2005.

[5] H. Honing. From time to time: The representation of timing and tempo. *Comp. Music J.*, 25(3):50–61, 2001.

[6] S.Schertenleib *et al.* Conducting a virtual orchestra. *IEEE MultiMedia*, 11(3):40–49, 2004.

[7] J. Segen, S. Kumar and J. Gluckman. Visual interface for conducting virtual orchestra. *ICPR '00*, page 1276, 2000.

[8] H. Katayose and K. Okudaira. Using an expressive performance template in a music conducting interface. In *NIME '04*, pages 124–129, 2004.

[9] F.Bevilacqua *et al.* Wireless sensor interface and gesture-follower for music pedagogy. In *NIME '07*, pages 124–129, 2007.

[10] S. Dixon, W. Goebl and G. Widmer. The "air worm": An interface for real-time manipulation of expressive music performance. In *ICMC '05*, pages 614–617, 2005.

[11] J.R.Jang, H.Lee and C.Yeh. Query by tapping: A new paradigm for content-based music retrieval from acoustic input. In *PCM '01*, pages 590–597. Springer-Verlag, 2001.

[12] F. Kurth *et al.* SyncPlayer - an advanced system for multimodal music access. In *ISMIR '05*, pages 381–388, 2005.

[13] S.Dixon and G.Widmer. MATCH: Music alignment tool chest. In *ISMIR'05*, pages 11–15, 2005.

[14] N.Hu, R. Dannenberg and G. Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *WASPAA '03*, pages 185–188, 2003.

[15] M. Müller, F. Kurth and T. Röder. Towards an efficient algorithm for automatic score-to-audio synchronization. In *ISMIR'04*, pages 365–372, 2004.