ORCHESTRAL ACCOMPANIMENT FOR A REPRODUCING PIANO

Christopher Raphael, Yupeng Gu
School of Informatics, Indiana Univ.
Bloomington, IN, USA
{craphael|yupgu}@indiana.edu

ABSTRACT

A system that generates flexible orchestral accompaniment of a computer-enabled piano is presented and demonstrated. We introduce a probabilistic model for the piano data that can be used for on-line and off-line estimation of the piano performance. The model is automatically trainable to the specific performer and piece under consideration. The on-line position estimates form the observable data for a trainable prediction engine that anticipates the future evolution of the performance. These ongoing predictions drive a phase-vocoded audio performance of the orchestra. We present results on a highly challenging gem from the Romantic piano concerto repertoire.

1. INTRODUCTION

Active research on Musical accompaniment systems continued for over two decades since the simultaneous premier of the first two such systems at the ICMC in 1984 [2], [6], Such systems seek to provide a flexible accompaniment to a live soloist that follow expressive timing and other performance nuances exhibited by the soloist. Our treatment of this problem is in a non-improvisatory concerto-like domain.

There are a variety of sensing mechanisms that an accompaniment system may use to understand the actions of the live player. The most widely applicable of these is, or course, audio since every musical instrument can produce audio. Schwarz [5] has a nice annotated bibliography detailing the last couple decades' work in this area. The results cited within include some impressive successes for wind, string and brass instruments in highly challenging domains, though polyphonic solo instruments such as the piano, pose exceptionally difficult and largely unsolved obstacles. Piano-specific issues making this scenario particularly hard include the complexity of the piano music and the ambiguity in mapping the score into the audio model. Some reasons for this ambiguity come from the score's inability to accurately represent offset times, due to pedaling and the "multi-channel" nature of the piano; the potential for some parts of the score to obscure other parts; and increasing similarity of the various chord signatures as note density increases.

The accompaniment problem is easier when symbolic music data are used, such as the output of a MIDI keyboard or wind controller. This domain has received some attention [1], as well as several commercial systems such as one recently released by Yamaha as a companion to the newest Disklavier. The disadvantage of symbolically-based score followers for "classical" music is that most instruments generating symbolic data lack the rich and varied depth of their acoustic counterparts.

There is, however, one significant exception to this generalization. Reproducing pianos, such as the Yamaha Disklavier and the Bösendorfer CEUS are rapidly gaining acceptance as research and performance tools. These instruments are *real* pianos with hammers, strings, etc., that are *also* capable of generating real-time performance data. Such instruments open the door to piano accompaniment systems by circumventing the challenge of audio score following for piano.

We address the problem of musical accompaniment for piano in a traditional concerto-type domain. Our work differs from past work in this domain through the coherent probabilistic model we present which can *learn* from examples. Through this model we can prescribe reasonable a decision process in the face of uncertainty. We model the listening problem with a dynamic Bayesian network whose observable variables are the timing and pitch information produced by the piano. Our system is able to follow a live player during *in vivo* experiments involving considerable rubato and error on the part of the soloist. We generate the real-time accompaniment for the pianist using a Kalman filter-type model to predict future musical evolution, while generating the audio accompaniment by phase-vocoding a prerecorded orchestra. We present experiments on the 1st movement of the Rachmaninov 2nd piano concerto using a Bösendorfer reproducing piano.

2. THE MODEL

The data we model here are a subset of the MIDI output from the reproducing piano. We denote these data as $y = (y_1, y_2, \ldots, y_N)$ where $y_n = (m_n, t_n)$ with m_n and t_n the midi pitch and time in seconds of the nth onset played by the piano.

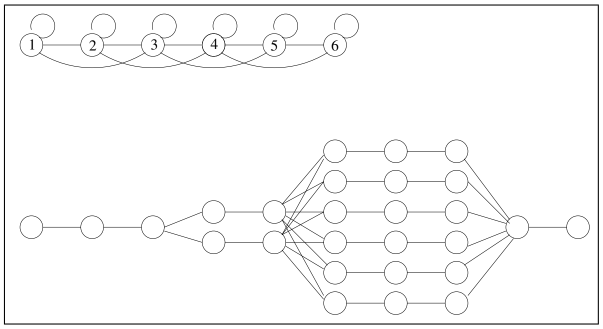

Figure 1. **Top:** A model showing possible transitions between the various notes of the musical score. Self-loops correspond to inserted notes while arcs that skip states correspond to deleted notes. **Bot:** A model showing how piano chords are handled. Since the order of observed notes in a chord is unknown, the model enumerates all possible orderings of the chord.

We seek a real-time correspondence between y and our musical score. To this end we define a hidden process $x = x_1, \ldots, x_N$ so that x_n indexes the note corresponding to y_n.

If a portion of our score is monophonic, we represent the score as a sequence of states as in the top panel of Fig. 1, one for each note. As indicated in this figure, we do not assume that the player necessarily plays every note on the score, nor do we assume that no accidental notes are added. Rather we allow both "self-loops" and "skips" in the graph, corresponding to inserted and deleted notes. Thus, for example, a path through this graph that begins by visiting states 1,3,3 and 4, corresponds to playing the 1st note, skipping the 2nd note, playing the 3rd note, adding an extraneous note between the 3rd and 4th notes, and playing the 4th note. Though we do not indicate this in the figure, we will allow state transitions that skip several notes as well.

The top panel of Fig. 1 oversimplifies due to the polyphonic nature of piano music, in which we may have several notes that begin at the same score time. If the player attempts to play these notes simultaneously, the actual order of onsets reported by the piano is essentially random. Thus we allow for all possible orderings of these events and model this as in the bottom panel of Fig. 1. For example, if k notes are to sound simultaneously, we have $k!$ possible ordering of these notes, so we will, in principle, include $k!$ sequences of k notes. Here, for the sake of clarity, we have omitted the self-loops and skips in the figure, though they are present in our actual model.

We model the transitions between the states probabilistically. We create an initial state for the graph, ξ_0, that precedes the first note(s) in our score and define $L(\xi_0) = 0$ where L gives the "level" of the state. The level of every other state is defined to be the number of transitions from ξ_0 needed to reach the state. We model the hidden sequence, x, as a Markov chain which begins in state ξ_0 and has transition probabilities

$$p(x_{n+1}|x_n) = q(L(x_{n+1}) - L(x_n))/B(x_{n+1}) \qquad (1)$$

for $n = 1, \ldots, N-1$. Here $\sum_{k=0}^{K} q(k) = 1$ and $B(x_{n+1})$ is the number of simultaneous notes in the chord of x_{n+1}. Thus $q(0)$ is the probability of the pianist adding an extra note, $q(1)$ is the probability of moving to the next score note, and $q(k)$ for $1 < k \leq K$ is the probability of skipping k notes. If we move across a chord boundary we must split the probability evenly between the $B(x_{n+1})$ various states at the same level, hence the divisor in Eqn. 1.

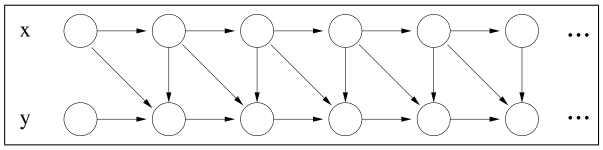

Figure 2. The model shows the joint dependency structure of the variables of our model. The top row is the sequence of hidden score positions while the bottom row denotes the (pitch,time) pairs that are observed from the piano.

We model the dependence of our observed piano data, y, on the state sequence, x, as

$$p(y|x) = p(y_1|x_1) \prod_{n=1}^{N-1} p(y_{n+1}|x_n, x_{n+1}, y_n)$$

The conditional dependence structure of this model is given in Fig. 2. We further assume that

$$\begin{aligned} p(y_{n+1}|x_n, x_{n+1}, y_n) &= p(t_{n+1}, m_{n+1}|x_n, x_{n+1}, t_n, m_n) \\ &= p(t_{n+1}|x_n, x_{n+1}, t_n) p(m_{n+1}|x_n, x_{n+1}) \end{aligned}$$

The distribution $p(m_{n+1}|x_n, x_{n+1})$ is modeled as

$$p(m_{n+1}|x_n, x_{n+1}) = \begin{cases} q^{\text{unif}}(m_{n+1}) & \text{if } x_n = x_{n+1} \\ q^{\text{mid}}(m_{n+1} - m(x_{n+1})) & \text{otherwise} \end{cases}$$

where $m(x_n)$ is the midi pitch associated with state x_n and q^{mid} is a probability distribution centering most of its mass around 0 and trailing away quickly as the argument increases above or decreases below 0. This expresses the notion that the player plays the correct note most of the time, and otherwise usually plays a nearby key. q^{unif} is a uniform distribution over the possible pitches on the piano, expressing the notion that if a note is inserted we really have no idea what note it will be.

The distribution $p(t_{n+1}|x_n, x_{n+1}, t_n)$ is modeled as

$$p(t_{n+1}|x_n, x_{n+1}, t_n) = \begin{cases} e(t_{n+1} - t_n; \lambda) & \text{if } x_n = x_{n+1} \\ g(t_{n+1}; \mu, \sigma^2) & \text{otherwise} \end{cases}$$

where $e(t_{n+1}; \lambda)$ is the Exponential density with parameter λ and $g(t_{n+1}; \mu, \sigma^2)$ denotes the Gamma density parameterized with its mean, μ, and variance, σ^2, rather than the usual shape and scale parameters. We then compute the mean and variance as $\mu(t_n.x_n, x_{n+1}) = t_n + A(x_n, x_{n+1})$ where $A(x_n, x_{n+1})$ is the expected time elapsed between the x_n and x_{n+1} and $\sigma^2 = \sigma^2(t_n.x_n, x_{n+1})$ is a fixed increasing function of $\mu(t_n.x_n, x_{n+1})$.

3. SCORE FOLLOWING

We accomplish score following by computing the *filtered* distribution on our score position x_n having observed the first n piano events, $p(x_n|y_1, \ldots, y_n)$. This is easily computed according to the recursion

$$p(x_{n+1} \quad | \quad y_1, \ldots, y_{n+1}) = \frac{\sum_{x_n} p(x_n|y_1, \ldots, y_n) p(x_{n+1}|x_n) p(y_{n+1}|x_{n+1})}{\sum_{x_n, x'_{n+1}} p(x_n|y_1, \ldots, y_n) p(x'_{n+1}|x_n) p(y_{n+1}|x'_{n+1})}$$

every time we observe a new piano event $y_{n+1} = (m_{n+1}, t_{n+1})$.

The filtered distribution expresses everything known about the current hidden state, so it seems reasonable to base any accompaniment decisions on this distribution. Unlike with score following of audio, we get observations at a non-uniform rate with our piano data. Thus, if the most recently received piano corresponds to a score position with high probability, we immediately conclude that the note has been played and take appropriate action.

We implement this as follows. If, upon updating the filtered distribution, we find that a single score position x^*_{n+1} has

1. $p(x^*_{n+1}|y_1, \ldots, y_{n+1}) > \tau$ for some threshold τ
2. the note x^*_{n+1} has not yet been reported
3. if x^*_{n+1} is a member of a chord then no other chord members have yet been reported

we determine that the note x^*_{n+1} has been played at the time we received y_{n+1}.

Once a piano note has been detected we change the playing of the accompaniment to be consistent with the detected note. The essential method we use is described compactly in [4], so we will explain this method only briefly here. We have a model of musical timing for the composite rhythm of solo and accompaniment parts, that is, the rhythm obtained by taking all solo and accompaniment onset times and arranging these in increasing order. The model is expressed by

$$\begin{pmatrix} p_{k+1} \\ s_{k+1} \end{pmatrix} = \begin{pmatrix} 1 & l_k \\ 0 & 1 \end{pmatrix} \begin{pmatrix} p_k \\ s_k \end{pmatrix} + \begin{pmatrix} \pi_k \\ \sigma_k \end{pmatrix} \qquad (2)$$

where p_k is the time of the kth composite event, s_k is the current tempo in seconds per quarter note, and l_k is the length of the kth composite event in quarter notes, and the $\{(\pi_k, \sigma_k)^t\}$ are independent, zero-mean, Gaussian random vectors.

When a new piano event has been observed, we treat the corresponding time variable, p_k in our model as having been observed. Then, conditioned on this new information, we compute the time of the pending accompaniment note. The accompaniment audio is generated by playing back a prerecorded audio file at variable rate, using phase-vocoding [3]. so that we reach the pending accompaniment note at the estimated time.

4. TRAINING THE MODEL

In our usual accompaniment scenario, the live player may play through the target piece of music many times before actually performing it. In many cases, the practice itself is the main focus, so there may never be a true "performance." Rather the accompaniment system is used as a practice tool. In either case, we take advantage of the information contained in past examples to automatically train our models, thereby adapting to the particular player and piece and achieving better results.

We perform two different kinds of training. The first is oriented toward improving the score-following, which views the timing of the piano part as a sequence of one- or many-note "chords" whose inter-onset intervals (IOIs) are modeled as independent Gamma random variables. As discussed above, the Gamma distribution is parameterized in terms of its mean and variance. When no training data are present, we compute the each mean IOI from the nominal local tempo and set the corresponding variance as a deterministic function of the mean. When training data are present, we compute the empirical mean and variance of each IOI, with a small bias in these calculations toward the initial values of these parameters. In either case, we compute the Gamma parameters from our estimated or hypothesized mean and variance — that is, we use the "method of moments." These estimates are updated after each rehearsal with the system.

Our accompaniment is generated by continually predicting the location of the pending orchestra note, given the times of the currently-identified piano notes and past orchestra notes, and setting the playback rate of our phase-vocoder to reach the pending accompaniment note at the predicted time. This prediction changes every time a new piece of information, such as a piano note detection, is made available. The accuracy of our predictions is important for creating a satisfying musical experience, especially in regions in which the player exercises significant liberty or *rubato*. We have found a high degree of commonality in the expressive timing exhibited by a particular player from performance to performance, even though the player's perception may be otherwise. Thus we can improve the performance of our system by learning the parameters of the $\{(\pi_k, \sigma_k)^t\}$ variables in the model of Eqn. 2. While these variables are initialized to have mean 0 with covariance matrix depending on the nominal length of the IOI, we adjust the mean vectors using maximum likelihood on the past performances. The computational aspects of this updating are somewhat involved, but are quite well known in the Bayesian belief network literature and are discussed in detail in [4] and [**?**].

5. EXPERIMENTS

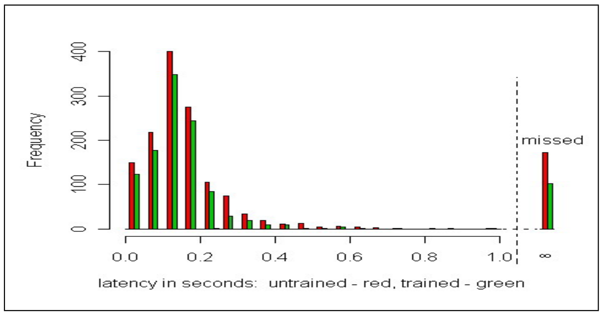

Figure 3. Histogram showing latency in note detection for the piano data before and after training the model.

We currently are performing experiments on the 1st movement of the Rachmaninov 2nd piano concerto. While in many scenarios MIDI-based score following might be rather straightforward, this piece poses significant challenges due to the sheer density of notes and potential for error in the "input." We have trained the score following model as discussed above using 4 rehearsals on complete performances of the movement. This training is especially useful in a piece such as the Rachmaninov, since *a priori* there is considerable uncertainty about note length making the following task harder. Training the IOI distributions leads to a more informed and better-functioning model.

We created ground truth for these 4 performances by hand-correcting the off-line score matches. This process found there were a total of 10,564 correctly played (possibly one-note) simultaneities. We trained and ran our algorithm on these performances using a "leave-one-out" training scheme and present results of our on-line detection before/after training, as percentages of the correctly played simultaneities. With our currently choice of τ we get 0.95%/0.30% incorrect detections and 1.62%/0.97% missed detections, though higher values of τ will shift notes from the former to the latter categories . Of the correct detections, 84.8%/88.7% occur immediately after the first note of the chord is received, while the latency on the remaining detections is given if Figure 3. The missed notes are contained in the right edge of the figure.

We also trained our prediction model as discussed above. Both video and audio of a complete performance of the movement with Yupeng Gu as featured soloist can be accessed at http://www.music.informatics.indiana.edu/papers/icmc09.

6. FUTURE WORK

We are currently working with piano faculty and students in the Jacobs School of Music (JSoM) at Indiana University. The JSoM has many piano students who possess the technically proficiency and musical maturity to serve as concerto soloist. It is unfortunate that most of these students will not have this experience while at IU, due simply to the small number of solo opportunities that are available — we have one student piano soloist perform with orchestra each semester. We are currently working on a demonstration in which three students will be featured as piano soloists in a public concert, accompanied by our program. We look forward to reporting on this event as it develops.

While this effort uses the MIDI data generated by the Bösendorfer piano, the high performance resolution data also created by the piano allows for interesting possibilities. This data includes 2 ms. measurements of key position for each key, allowing one to *anticipate* the strike time of the piano notes. This information is especially valuable in situations where the piano note cannot be accurately predicted, such as a simultaneous entrance of both piano and orchestra. At present, our system simply waits until the piano has been detected before launching the orchestra, thus guaranteeing the orchestra will be late. However, we believe the information will be more generally useful, by offering earlier knowledge of the performer's actions.

7. REFERENCES

[1] J. Bloch and R. Dannenberg, "Accompaniment of keyboard performances," in *Proc. Int. Comp. Music Conf.* Int. Computer Music Assoc., 1985, pp. 279–289.

[2] R. Dannenberg, "An on-line algorithm for real-time accompaniment," in *Proceedings of the International Computer Music Conference, 1984*. Int. Computer Music Assoc., 1984, pp. 193–198.

[3] J. L. Flanagan and R. M. Golden, "Phase vocoder," *Bell System Technical Journal*, pp. 1493–1509, Nov. 1966.

[4] C. Raphael, "A bayesian network for real-time musical accompaniment," in *Advances in Neural Information Processing Systems, NIPS 14*, T. Dietterich, S. Becker, and Z. Ghahramani, Eds. MIT Press, 2002.

[5] D. Schwarz, "Score following commented bibliography," IRCAM, Tech. Rep., 2003. [Online]. Available: http://www.ircam.fr/equipes/temps-reel/suivi/bibliography.html

[6] B. Vercoe, "The synthetic performer in the context of live performance," in *Proceedings of the International Computer Music Conference, 1984*. Int. Computer Music Assoc., 1984, pp. 199–200.