Music Plus One: A System for Expressive and Flexible Musical Accompaniment

Christopher Raphael*
Department of Mathematics and Statistics
University of Massachusetts, Amherst
raphael@math.umass.edu

Abstract

We describe a computer system for musical accompaniment in non-improvisatory music and present a comparison with the commercial system SmartMusic^{TM}.

1 Introduction

Our ongoing work, "Music Plus One," develops a computer system that plays the role of musical accompanist in a piece of non-improvisatory music for soloist and accompaniment. The system takes as input the acoustic signal generated by the live player and constructs the accompaniment around this signal by combining the constraints of the musical score with learned musical interpretations for both the solo and accompaniment parts.

There have been several other research efforts addressing this goal including Dannenberg [1], Bloch and Dannenberg [2], Dannenberg and Mukaino [3], Grubb and Dannenberg, Vercoe [5], and Vercoe and Puckette [4]. The main difference between our work and the cited research is the fundamental emphasis we place on probabilistic modeling; there are two principle examples of this. Firstly, the method we use to track the live performer is based on hidden Markov models (HMMs). The principle advantages of an HMM-based approach include fast dynamic programming algorithms, automatic adaptation, and quantification of uncertainty, as we will discuss. The second example of probabilistic modeling is the way we try to mediate between the goals of accurate synchronization and expressive playing. We construct an explicit probabilistic model that represents the basic asymmetry between the soloist (the leader) and the accompanist (the follower). The model combines the information provided by the score, the on-line analysis of the soloist's performance, and training from past performances of both the solo and accompaniment parts into a coherent generative model. Thus the output of our accompaniment system is the most likely accompaniment performance given all currently available information.

*This work is supported by NSF grant IIS-9987898.

2 Modeling

2.1 Listen

We have partitioned the accompaniment problem into two components, *Listen* and *Play*. Listen takes the soloist's acoustic signal as input and performs a real-time analysis of the signal. The output of Listen is essentially a running commentary on the acoustic input that identifies note boundaries in the solo part; these identifications are communicated with variable latency. The Listen component must satisfy several objectives:

1. The analysis must be computationally efficient in order to satisfy the real-time demand of our system. In our current system the signal accumulates at a rate of 8000 samples per second.
2. The approach must be automatically trainable to adapt to new acoustic environments. Variables in this environment include the choice of instrument, the player of the instrument, the placement of the microphone, the choice of accompaniment instrument (our system hears the accompaniment too), and the acoustics of the room.
3. The method must be accurate. Some musical events such as octave slurs and rearticulations are difficult to detect initially, while their locations become increasingly clear with the benefit of more hindsight. To achieve accurate boundary detections the system must delay the detection of a note until enough data are available to estimate the note position accurately.

The backbone of our implementation of Listen is the hidden Markov model (HMM) methodology so successful in speech recognition. HMMs provide a means of satisfying the three objectives we mention as follows.

Computation with HMMs is based on dynamic programming algorithms which provide the speed necessary for our real-time application. In particular, a modification of the "forward" algorithm leads to a fast and accurate on-line analysis algorithm for identifying note boundaries.

Our HMM contains many parameters such as those that describe what one expects to hear when a particular note is sounding and those describing the probabilistic evolution of the solo part. It is neither feasible nor desirable to set so many parameters by hand. However, HMMs provide a method for unsupervised training in which the many parameters of the model are set using a collection of signal, score pairs. The virtue of this approach is that it is not necessary to match the scores to the actual data by hand. Rather a probabilistic match evolves through course of the training algorithm. Thus HMMs bring a much needed adaptability to our system.

Finally, the probabilistic nature of the HMM allows us to measure our uncertainty about the position of the live player at any given time in terms of a probability distribution. Thus our system waits until an event is in the past with desirable degree of confidence before estimating the position of that event. Such a capability is especially important in analyzing musical signals which can be locally ambiguous, but become relatively easy to label when more information becomes available. We have found this aspect of HMMs to be essential in an application that requires both accuracy and low latency in detection — *in that order*. A complete description of our implementation of Listen is contained in [6].

2.2 Play

The challenge of the *Play* component is to construct the accompaniment which, at the lowest level, is a series of MIDI events events dispatched by our program to a sound module or synthesizer. As with the human musical accompanist, the music produced by our system must depend on a number of different knowledge sources. From a modeling point of view, the primary task is to develop a model in which these disparate knowledge sources can be expressed in terms of some common denominator. We describe here the three knowledge sources we use.

We work with non-improvisatory music so naturally the musical score, which gives the pitches and relative durations of the various notes, must figure prominently in our model. The score should not be thought of as a rigid grid prescribing the precise times at which musical events will occur. Rather, the score gives the basic elastic material which will be stretched in various ways to produce the actual performance. The score simply does not address most interpretive aspects of performance.

Since our accompanist must follow the soloist, the output of he Listen component, which identifies note events in the solo part, is also a central knowledge source. However, since there will be variable latency in the detection of musical events we believe strongly that any successful accompaniment system cannot synchronize in a purely responsive way. Rather it must synchronize by anticipating the solo part's future evolution using past data, as human musicians do.

While the same player's performances of a particular piece will vary from rendition to rendition, there is much to be learned by studying past examples; this is, of course, one of the main reasons musicians rehearse together. These examples, both of solo performances and human renditions of the accompaniment part constitute the third knowledge source for our system. These knowledge sources embody two types of information: the soloist and accompaniment interpretations. The solo data are used primarily to teach the system how to predict the future evolution of the solo part (and to know what can and cannot be predicted reliably). The accompaniment data are used to learn the musicality necessary to bring the accompaniment to life.

We have developed a probabilistic model, a Bayesian belief network, that represents all of these knowledge sources through a jointly Gaussian distribution containing hundreds of random variables. The observable variables in this model are the estimated soloist note onset times produced by Listen and the times at which the accompaniment notes are played. Between these observable variables lie several layers of hidden variables that describe unobservable quantities such as local tempo, change in tempo, and rhythmic stress.

The network consists of two basic parts, one for the accompaniment and one for the soloist. Initially we create a probabilistic model for the time evolution of each of the two parts. In both cases we use a model in which a state variable representing time and tempo evolves as we move through the sequence of notes.

To be more specific, we model the time evolution of the solo part as follows. For each of the solo notes, indexed by $n = 0, \ldots, N$, we define a random vector representing the time, t_n, (in seconds) and the "tempo," s_n, (in secs. per measure) for the note. We assume that this sequence of random vectors evolves according to

a random difference equation:

$$\begin{pmatrix} t_{n+1} \\ s_{n+1} \end{pmatrix} = \begin{pmatrix} 1 & l_n \\ 0 & 1 \end{pmatrix} \begin{pmatrix} t_n \\ s_n \end{pmatrix} + \begin{pmatrix} \tau_n \\ \sigma_n \end{pmatrix} \quad (1)$$

$n = 0, \ldots, N-1$, where l_n is the musical length of the n^{th} note in measures (e.g. a quarter note in 6/8 time is 1/3 of a measure) and the $\{(\tau_n, \sigma_n)^t\}$ are independent Gaussian random vectors.

The distribution of the $\{\sigma_n\}$ tend to concentrate around 0 which expresses the notion that tempo changes are gradual. The means and variances of the $\{\sigma_n\}$ show where the soloist is speeding-up (negative mean), slowing-down (positive mean), and tell us if these tempo changes are nearly deterministic (low variance), or quite variable (high variance). The $\{\tau_n\}$ variables describe stretches (positive mean) or compressions (negative mean) in the music that occur without any actual change in tempo. Thus, the distributions of the $(\tau_n, \sigma_n)^t$ vectors characterize the solo player's rhythmic interpretation. Both overall tendencies (means) and the repeatability of these tendencies (covariances) are expressed by the distributions of these vectors.

We define an analogous model for the accompaniment part and then both models, solo and accompaniment, are trained from actual performance data using the EM algorithm. We refer to this accompaniment distribution as the accompanist's "practice room" distribution since we train this model from MIDI performances of the accompaniment playing alone. Typically we train the solo model from solo performances collected during a rehearsal phase. These performances are analyzed with an "off-line" version of our Listen module to determine note onset times and these data form the basis of our solo training.

The solo and accompaniment models are then combined into a network containing both parts. We build this joint network in a way that captures the notion that the accompaniment must follow the soloist. In particular, at points of synchronicity between the two parts, the accompaniment will essentially "inherit" the time and tempo of the soloist. The remaining accompaniment notes are modeled to come from their conditional distribution given the synchronous notes, where this conditional distribution is computed under the learned accompaniment practice room distribution. Thus the accompaniment's first duty is to the soloist, while any interpretive issues not forced by the need to synchronize will be resolved through the practice room model.

The methodological key to our real-time accompaniment algorithm is the computation of (conditional) marginal distributions facilitated by the ideas from the Bayesian Belief Network paradigm. Our scheduling mechanism concentrates on the problem of determining when the pending accompaniment note should be played. At any point during the performance some collection of past solo onset estimates (from Listen) and past accompaniment note times will have been observed. Conditioned on this information we can compute the distribution on the next unplayed accompaniment note. Essential to our system's musical well-being is the fact that this distribution depends on all three knowledge sources we have introduced into our model: the musical score, the musical training of the solo and accompaniment parts, and the solo note times estimated by Listen. We then schedule the pending note according to this conditional distribution, for instance, at the mean time. If a new solo note is detected before the pending accompaniment note is actually played, we recompute the scheduled time for the pending accompaniment note as before to account for the new information available to us. Thus a note's time might be scheduled several times before it is actually played in such a way that, at the time it is finally played, we have the benefit of all currently available information.

A more detailed discussion of this work can be found in [7]. Several examples of our current state of the art can be heard at: http://fafner.math.umass.edu/music_plus_one.

3 Comparisons

We offer here a numerical comparison between our system, Music Plus One, and the commercial system SmartMusic™ produced by Coda Music Technology. Comparison between musical accompaniment systems is somewhat problematic due to the many different axes on which success can be measured. We focus here on comparing the degree to which the systems synchronize with the soloist. While we acknowledge that an accompaniment system should have a broader agenda than merely trying to synchronize with the soloist, the measurement of success in these more musical goals is subjective. We are now in the process of enlarging our study to include an analysis of these subjective components through a user-study, and including more systems into the study. However, we content ourselves here with the more straightforward and objective analysis of synchronicity.

A meaningful comparison requires the systems to be run with identical input, and for this reason we used a prerecorded data set consisting of 8 performances of the opening section of Robert Schumann's 2nd Romance for Oboe and Piano. This piece was chosen since, due to its free and expressive nature, it represents a significant challenge to an accompaniment

system's ability to follow a soloist. These data were recorded with the soloist playing alone. Before any experiments were done, one of these performances was chosen to be the test case, while the others were reserved as training.

Each system was tested under two different experimental conditions we refer to as *sightreading* and *rehearsed*. In the sightreading experiment each system is given only two pieces of information: the initial tempo and a series of breakpoints. These breakpoints are places at which the system waits until it hears input from the soloist before beginning to play, such as after a *fermata*. We used the same tempo and breakpoints for each of the two systems and the latter were those suggested by SmartMusic™.

We believe that the fundamental description of the accompaniment problem should include rehearsal data, since this information is generally available and indispensable to the human accompanist. For this reason, the role of rehearsal data figures prominently in the basic probabilistic model we construct. SmartMusic™ does not have a direct mechanism for using rehearsal information, but does allow the user to tap in beats over any section of the music. In an effort to facilitate a comparison under the rehearsed condition, we tapped in beats for two of the training examples in preparation for our rehearsed experiment with SmartMusic™. The performance of SmartMusic™ did not depend significantly on which example was used. In contrast, Music Plus one uses a collection of training performances which were the complete data set minus the test case.

In comparing the systems we recorded both the accompaniment and the input on two different channels of an audio file. In the case of SmartMusic™ these were analyzed separately by program to produce estimates of times at which each solo note and accompaniment note occur; these were later corrected by hand. In analyzing Music Plus One, the exact times of accompaniment notes were known, so we only needed to estimate solo note times, as above. For each system we computed the difference in note onset for every pair of coincident solo and accompaniment notes. The evolution of these errors is shown in Figure 1 in which error (in seconds) is plotted against score position (in measures). In both cases SmartMusic™ loses sight of the soloist and drifts away. Audio files of the four performances can be heard at http://fafner.math.umass.edu/icmc2001.

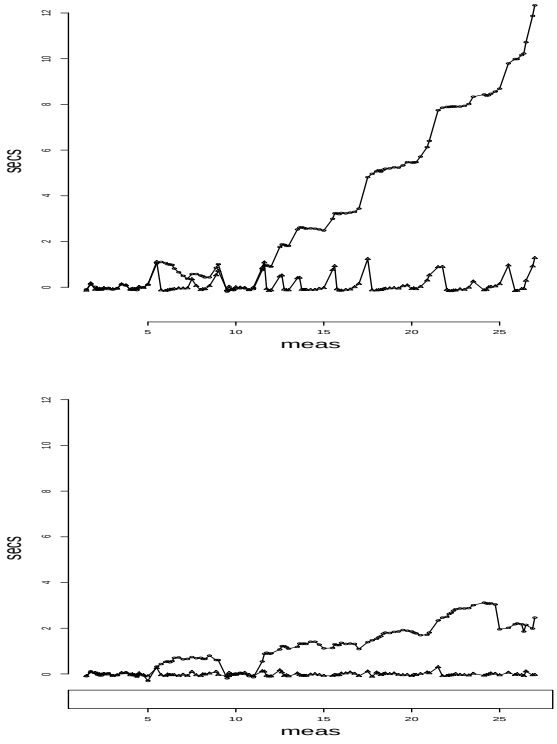

Figure 1: Synchronization errors under sightreading (Top) and trained conditions (Bottom).

References

[1] R. Dannenberg , "An On-Line Algorithm for Real-Time Accompaniment," *Proceedings of the ICMC, 1984* IRCAM Paris, France, 193–198, 1984.

[2] J. Bloch, R. Dannenberg, "Real-Time Computer Accompaniment of Keyboard Performances," *Proc. of the ICMC, 1985* 279–289, Burnaby, British Columbia, Canada, 1985.

[3] R. Dannenberg, H. Mukaino "New Techniques for Enhanced Quality of Computer Accompaniment" *Proc. of the ICMC, 1988* 243–249, Köln, 1988.

[4] B. Vercoe, M. Puckette "Synthetic Rehearsal: Training the Synthetic Performer," *Proc. of the ICMC, 1985* 275–278, Burnaby, British Columbia, Canada, 1985.

[5] B. Vercoe, "The Synthetic Performer in the Context of Live Performance," *Proc. of the ICMC, 1984* 199–200, IRCAM Paris, France, 1984.

[6] Raphael C. (1999), "Automatic Segmentation of Acoustic Musical Signals Using Hidden Markov Models," *IEEE Trans. on PAMI,* Vol. 21, No. 4, pp. 360–370.

[7] Raphael C. "A Probabilistic Expert System for Automatic Musical Accompaniment," to appear in *J. of Comp. and Graph. Stats.*