Automatic Accompaniment

Gerry J. Kim, POSTECH, Korea
Visiting Scholar @ IMSC, USC

What is Automatic Accompaniment?

To synchronize a machine performance of music to that of a human

- In computer accompaniment, it is assumed that the human performer follows a composed score of notes and that both human and computer follow a fully notated score
- In any performance, there will be mistakes and tempo variation, so the computer must listen to and follow the live performance, matching it to the score
- AKA score following, synthetic performer, intelligent performer, intelligent accompanist, …
- (Chord accompaniment)

Today

- Issues in Automatic Accompaniment
- Early Work by R. Dannenberg (Reactive)
  - Matcher: Basic algorithm
  - Accompanist: Dealing with tempo variation
  - Extensions
- More Recent Work by C. Raphael (Predictive)
  - Listener / Pitch Detection: Hidden Markov Model
  - Synthesis: Phase Vocoder
  - Anticipate: Probabilistic knowledge fusion
- Fakeplay
- Discussion

Demo: Casio CTK Series ($149.99 only!)
(Yamaha has something similar, too)

- Piano teaching
  - Step 1: Timing (Accompaniment waits)
  - Step 2: Notes (Accompaniment waits)
  - Step 3: Normal (Accompaniment proceeds)
- Chord playing
  - Play bass and chord accompaniment in accordance to the finger(s) designation
  - Rhythm selection and variation
  - Fill in
  - Intro and Ending
Important issues in Automatic Accompaniment

- Tracking the solo (performer)
- Matching the solo and accompaniment
- Real time requirement
- Performance and improvisation
  - Learning the patterns of the solo and accompaniment

Dannenberg (1984-): CMU

- (1984: Vercoe @ MIT)
- 1984: Accompaniment to monophonic solo
- 1985: Extension to polyphonic solo (with Bloch)
- 1988: More extensions (with Mukaino)
- 1994-: Accompaniment for ensemble/voice (with Grubb)
- ... 1997-2002: Improvisational music / Learning styles of music (with Thom)

Introduction

- System listens to one or more performance (input, called solo)
  - Term: solo score → machine readable format of performance
- Compare solo performance to stored scores
  - Assume high correlation between performance and (stored) score
  - (No Improvisation)
- System synthesizes accompaniment
  - Deals with tempo and articulation change
  - Outputs accompaniment score
  - Time in (stored) score → virtual time
    - Warped into real time to match tempo deviation in solo performance
    - E.g. Virtual time in Score: 0 100 110 ...
    - Mapped and transformed into: (10000 10100 10200 ...)
    - Adjusted in accordance to solo input: (9995 10007 ...

Overview: Four Components

- Input preprocessor: extracts information about the solo from hardware input devices (e.g. pitch detector, keyboard, ...)
- Matcher: reports correspondences between the solo and the score to accompanist
- Accompanist: decides how and when to perform the accompaniment based on timing and location information it receives from the matcher
- Synthesis: hardware and software to generate sounds according to commands from accompanist
Matcher

- Compares solo to score to find the best association between them
  - Can consider number of things like pitch, duration, etc.
  - Here, only uses pitch information
    - (And the timing factor for polyphonic case)
      - to group simultaneous notes into one event
  - Events in the score are totally ordered
- Must tolerate mistakes
- Produce output in real time

Monophonic matcher

- Compute: rating of association between (on-going) performance and score:
  - Maintain a matrix where row corresponds to score events and column corresponds to (on-going) performance events
    (new column is computed every time new performance events occur)
- Observation: rating is monotonically increasing based on prior results (dynamic programming)
  - Maximum rating up to score event r and performance event c will be at least as great as one up to r-1 and c because considering one more score event cannot reduce the number of possible matches
  - At least as great as the one up to r, c-1 where one less performance event is considered
  - If score event r matches performance event c, then the rating will be exactly one greater than one up to r-1, c-1
- Whenever match results in larger value (max. rating up to that point) then report that the performer is at the corresponding location in score

Algorithm

```
for all i, maxrating[i,-1] = 0;
for all j, maxrating[-1,j] = 0;
for each new performance event \( p[c] \) do
  begin
    for each score event \( d[r] \) do
      begin
        maxrating[r, c] = max(maxrating[r - 1, c],
                          maxrating[r, c - 1]);
        if \( p[c] \) matches \( d[r] \) then
          maxrating[r, c] = max(maxrating[r, c],
                          1 + maxrating[r - 1, c - 1]);
      end
  end
```

Dynamic programming: Only previous column is saved (not entire matrix)

Reporting the match

```
performance:
A G E D G B C
score:
A 1 1 1 1 1 1 1
G 1 2 2 2 2 2 2
E 1 2 3 3 3 3 3
G 1 2 3 3 4 4 4
A 1 2 3 3 4 4 4
B 1 2 3 3 4 5 5
C 1 2 3 3 4 5 5
```
Polyphonic case

- A G E can be coming in as (G E A), (A E G) ...
  - But within fraction of seconds ...
    - Grouping incoming performance as one event
      - Static grouping
      - Dynamic grouping

- What does it mean to have the best association between group of notes vs. group of notes?

- When to report the score location?

Back to maxrating function (1)

- p[i]  
  - ith event in performance
- p[1:i]  
  - first i events in performance
- s[j]  
  - jth event in score
- s[1:j]  
  - first j events in score

- Find j such that by some criteria:
  
  \[
  \text{maxrating}(p[1:i], s[1:j]) \text{ is the maximum when given } i \text{ performance notes}
  \]

Back to maxrating function (2)

maxrating tries different associations (in theory) and compute match value

- Label performance symbols extra, wrong, or right
- Label score as missing (if needed)
- Compute the following value:
  
  \[
  \text{length of score prefix} - C_1 \text* number of wrong notes} - C_2 \text* number of missing notes - C_3 \text* number of extra notes
  \]

  - There can be several j values for which max. rating occurs
  - Tie breaker: use one with smallest value

Reporting the matches

Match is good enough to report when the last note of the performance is consistent with the score position which was most likely on the basis of the previous performance history (when rating increases)
Using the maxrating function (monophonic case)

Previous algorithm is actually implementation of above for the simple monophonic case.

<table>
<thead>
<tr>
<th>Performance</th>
<th>Score</th>
<th>performance:</th>
<th>score:</th>
</tr>
</thead>
<tbody>
<tr>
<td>A G E D A G B C</td>
<td>A G E D G B C</td>
<td>A G E D G C</td>
<td>G E D G C</td>
</tr>
<tr>
<td>Try association given new D.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-A, G-G, E-E, D-G.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-G, G-E, E-extra, D-G</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Let’s say the maximum rating did not improve from last match (no report)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Then, new performance event A came in: A G E D A vs. A G E G A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-A, G-G, E-E, D-missing, A-A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(perhaps the rating improved compared to last rating value, record)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

By the characteristics of the matching function (can be computed from previous match, it can be implemented in dynamic programming fashion as illustrated in the algorithm description)

Grouping the notes (polyphonic case)

- **Static**: parse solo performance into compound musical events (called cvts) and treat it as one event
- **Group series of notes within some threshold as one group (because in reality slight timing diff. among simul. hit chord)**
  - 8 16th notes per second (played fast): 125 msec between notes
  - So use 90-100 msec as threshold
  - But what about rolled chord?
    - If a note is much closer than to the previous note than it is to the predicted time of the next solo event that it is declared to be in the same cvt even if not within the threshold (if the time between two notes is less than some fraction of predicted time from the first event to the next cvt, the second note is grouped with the first (use _ here)
      - This value is related to the limit on how much faster the soloist can play than the accompanist thinks he is playing

Few details (polyphonic case)

- Parse the stored score also
- **When do we apply match process?**
  - Each time we process a solo event and update the values when another note is assigned to the same cvt
  - Tentative match before the notes for the whole chord is played is still reported (not so important ...)
  - When new solo event comes in not part of previous cvt the last best match is declared correct
  - Given matches up to previous score, interim match between unfinished performance chord and cvt in score (partial match)
    - if performed events in score cvt = (if of performed events not in score event / performed events)
    - > 0.5 it is a match

Dynamic grouping

- **Considers all possible grouping of performance events independent of timing in order to achieve best association**
- **See paper**
- Both static and dynamic grouping works reasonably well in practice
Accompanist

- Matcher reports the solo event
- Its real time is reported
- Its virtual time is identified from the match
- The relationship between the virtual time and the real time is maintained to reflect it as tempo
  - Virtual time space is linear function of real time (i.e. slope = tempo)
- Direct jump according to new tempo can produce strange performance (sudden fast-forwarding or repeating certain notes)
  - Tempo change must be have the right reference point

Accompanist (2)

- When a match occurs:
  - Change virtual time of currently played note
    - If difference is less than threshold → deemed correct (not tempo change)
    - Subtle articulation possible (demo with Fakeplay later)
    - Accompaniment was lagging
      - Quickly play up to new virtual time (= real time of matched solo)
      - If dramatic change (time difference of > 2 sec), just go there without playing intermediate skipped notes
      - This happens when solo mistakes the long rest ...
    - Accompaniment was ahead
      - Continue to play current note until its end (while solo catches up presumably)
  - Change the clock speed for future playing
    - Use last few matches and the time differences to maintain current tempo (circular buffer)

Extensions

- Multiple matchers (competing hypothesis)
- Special notes: Trills and Glissandi
- Making it faster using bit vectors for implementation
- Learning the player’s (solo’s) style → predict when improvising and provide suitable accompaniment
- MIR / Style Recognition
- Coda Music Smartmusic Software:
  - Thom (Demo)

Raphael (1998-)

- Oboe player (Winner of S.F. Young Artist Competition) and Professor of Mathematics at Umass
- “Music minus one” project:
  - Solo tracking using HMM (Listener)
    (different from pure pitch detection)
  - Probabilistic approach to prediction (Player)
    - Reflect solo and accompanist’s expression
      - More musicality
      - Better synchronization
    - Based on prior performances (rehearsals)
      - Continuous update of the model during performance (Learning)
    - Use actual recording for accompaniment (sounds better?)
Listen (1): Solo segmentation problem

- Divide solo signal into short frames
  - About 31 frames per second
  - Goal: label frame with score position

- For each note in solo, build a small Markov model
  - States
    - Associated with pitch class and portions of notes (attack, sustain, rest)
    - Variations by types of notes
    - Flexible to allow length variation
  - Chain individual note models to form a model for whole score
    - Markov model with state X’s
    - Transition probabilities
    - Chosen based on average and variance of note length
    - Can be trained, too (Several performances + Update Algorithm)

Markov Model

- States (in time): $w_i(t)$

- Transition probabilities: $P(w_{i+1} | w_i(t)) = a_{ij}$
  - First order \( \rightarrow \) current state only dependent on previous one

- What is probability of having a sequence, say $w_1(1), w_2(2), w_3(3), w_4(4), w_5(5)$?
  \( \rightarrow a_{23} a_{31} a_{15} a_{51} \) (Forward-Backward Alg.)

- Hidden Markov Model: The states we are interested in are only indirectly observable, with another probability distribution

What can we do?

- Evaluation:
  What is probability of a sequence, $w_1(1), w_2(2), w_3(3), w_4(4), w_5(5)$?
  \( \rightarrow a_{23} a_{31} a_{15} a_{51} \) (Forward-Backward Alg.)

- Decoding (HMM): Given observations, what is the most likely (hidden) states that produced this?
  \( \rightarrow \) Greedy search (may not produce feasible sequence)

- Learning: Figure out the transition probabilities from training samples
  \( \rightarrow \) Baum and Welch Algorithm
Listen (2): HMM

- **Hidden Markov Model**
  - Hidden (true) states (which point in score) of a given situation
  - The HMM amounts to identifying the transition probabilities among the states
    \[
    P(X = x) = P(X_0 = x_0) \prod_{k=1}^{K-1} P(X_k = x_k | X_{k-1} = x_{k-1})
    \]
  - Observable output
    - Another probability distribution exist for these outputs given true state
    \[
    P(Y = y | X = x) = \prod_{k=1}^{K-1} P(Y_k = y_k | X_k = x_k)
    \]

Listen (3): Training the HMM

- What we get is acoustic feature data, Y, for each frame (freq, amplitude, etc).
- P (Y I X) can be learned by Baum-Welch algorithm with training data unrelated to the piece.

Estimating starting times of M not

Decoding: Give me most probable sequence of X’s (frame by frame note labeling), given a solo performance
- from this state sequence, we can obtain on set times of notes

Q: What if I am wrong once in a while?
- The nature of decoding algorithm is local optimization, thus can result in a sequence that is not “allowable” (note 1→ note 3) which is ok for us because unallowable sequence (e.g. skipping a note) actually happened! (complicated stuff)

Figure 1: A Markov model for a note allowing an optional silence at the end.
Estimating starting times of M notes

- For real time purpose (where we do this incrementally as we hear the performance) we wish to know when each note on set has occurred and we would like this information “shortly” after the onset has occurred ...
- Assume we detected all notes before note m and currently waiting to locate note m. We examine successive frames k until

\[ P(X_k \geq \text{start}(m)) | Y_1 = y_1, Y_2 = y_2, \ldots, Y_k = y_k \] > \tau

Then, assuming note m has happened, compute

\[ \hat{k}_m = \arg \max_{k \geq k_m} P(X_k = \text{start}(m)) | Y_1 = y_1, Y_2 = y_2, \ldots, Y_k = y_k \]

Indexing the Audio Accompaniment Part

- Mostly similar to segmenting the solo part
- Use the score the represent it as series of chords
  - By virtue of containing certain notes, it exhibits certain frequency characteristics
  - The polyphony makes applying training by B-W algorithm difficult
  - Construct P(Y|X) by hand based on frequency characteristics
  - (given a chord, figure out joint probability distribution for frequency bands ... ?)

Synthesize: Phase Vocoder (Demo)

- Divide the signal into sequence of small overlapping windows
- Compute the Fourier transform for each window
- Magnitude and phase difference btw. consecutive windows saved for each frequency band (intended mainly for sounds with few partials)
- Nth frame of output is replayed using inverse FT with saved magnitude and the accumulated phase function (time domain function reconstructed while retaining its short-time spectral characteristics) implies preserving the pitch, and avoiding the ‘slowing down the tape’ pitch drop.
  - The modified spectrogram image have to be 'fixed up' to maintain the duration/distance of the original, thereby ensuring the correct alignment of successive windows in the overlap-add reconstruction

Anticipate: Baysian Belief Network

\[
\begin{pmatrix}
  f_{n+1} \\
  \tau_{n+1}
\end{pmatrix} =
\begin{pmatrix}
  1 & 0 \\
  \frac{t_n}{\tau_e} & 1
\end{pmatrix}
\begin{pmatrix}
  f_n \\
  \tau_n
\end{pmatrix} +
\begin{pmatrix}
  \tau_n
\end{pmatrix}
\]

- t: onset time for solo/accompaniment note
- s: tempo of solo/accompaniment
- -\(\tau_e\) noise (zero mean) \(\tau\) variation in rhythm
- -\(\tau_e\) noise (zero mean) \(\tau\) variation in tempo
- l: musical length of nth note
- \(l_n = m_{n+1} - m_n\)
- m’s are various note positions (obtained from Listener/Player)
Belief Network: Modelling Causality

(HMM is a kind of BN)

Training the network

- Learns the $\cdot \cdot \cdot$ and $\cdot \cdot\cdot$
  - $\approx$ Message passing algorithm
  - $\approx$ EM algorithm (Baum-Welch)
- Run the learning algorithm with accompaniment only
- Then run with solo performance (solo overrides only where solo and accompaniment overlap) $\rightarrow$ keep accompaniment only expression but still follow the solo
- Q: What about $\cdot \cdot \cdot$ and $\cdot \cdot\cdot$? $\rightarrow$ obtainable from training samples?

The Belief Network

$\alpha_n = t_n + \alpha_{n-1} \left( \begin{array}{cc} 1 & 1 \\ 0 & 1 \end{array} \right) \left( \begin{array}{c} t_n \\ \alpha_{n-1} \end{array} \right)$

Accompaniment generation

- At any point during the performance, some collection of solo notes and accompaniment will have been observed
- Conditioned on this information, we compute the distribution on the next unplayed accompaniment event (only the next one for real time purpose)
- Play at conditional mean time (and reset play rate of vocoder)
Pros and Cons

- Reactive
  - Limits to what can be done in real time

- Predictive
  - Learning (how many rehearsals? ~10)
  - Individualized
  - Performance as good as they claim? (you be the judge)

Fakeplay (1999-)

- Focused on appreciation and enjoyment
  - Active (playing): Talent, practice, organization, but deeper sense of enjoyment (dance, hum, tap, air-guitaring (vs. Passive)

- Q: Is there a way for "the musically less fortunate (like me!)" to somehow experience, at least partially, enjoyment from active performance
  - Partial participation (interact with music or another player)
  - Ownership of control
  - Replicate the "aura"
  - Free people from the "Skill Problem"
  - Implication for Music Education

Experts vs. Non-Experts: Skill Problem Issue

- Learning or appreciation is best achieved when user is in control
- But control needs skills (long years of practice ...)
- Experts can concentrate on musical expression (has cognitive room) – will multimodality be a distraction?
- Non-experts concentrate more on musical information

For appreciating both music and additive elements introduced by a new breed of music environments, we must free the user from worrying about hitting the wrong key ...

BeatMania (1999 ?)
Demo

Control Interface: Air-Piano

- Direct Interaction (Rhythm)
  - Need to play through "concrete" interaction, not just lead
- Which music parameters (to ensure sense of participation)
  - Progression: Conducting (Lookahead/Delay Problem)
  - Intensity / Accent
  - Tempo
  - Duration (Rubato)

"Conductor-player in a Concerto"

Air-Piano: Minimal Piano Skill

MIDI File Parsing
- Construct MIDI Event Linked List
- Postprocessing

Check user input
- Tap
- Intensity change
- Tempo change

Compute "Play time"

Examine Linked List and play events up to events whose time stamp < "Play time"

Update graphics
New Directions

- Music cognition and perception
  - Meter, Rhythm, and Melody Segmentation
  - Performance and Improvisation
  - MIR
- Performance
  - New Interfaces and Ergonomics
  - New Environment
    - VR / AR / Ubicomp
- New interfaces for music control
  - Gesture
    - Instrumental
    - Dance oriented
    - Conductor
  - Control Surface Accessibility
    - Usual
    - Virtual
  - Affective (Physiological data ?)
  - Novel
Answer: VR and Computer Music?

<table>
<thead>
<tr>
<th>Performer</th>
<th>intent</th>
</tr>
</thead>
<tbody>
<tr>
<td>central nervous system</td>
<td></td>
</tr>
<tr>
<td>motor system</td>
<td>proprioception</td>
</tr>
</tbody>
</table>

Virtual world itself acts as source of multimodal feedback and a place for interaction (vs. virtual music instrument)

Closed loop system of a virtual music environment (Inspired by music performance model by Pressing)

Highly “Present” Virtual Musical Environment

DNA: An Immersive Virtual Music Band (SIGGRAPH 97)

DIVA: An Immersive Virtual Music Band (SIGGRAPH 97)

Iamascope: An Immersive Virtual Music Band (SIGGRAPH 97)
Well, they’re interesting and nice, but …

- Not for the general public (Skill problem not solved)
- Not oriented for performances of known scores which is the most typical type of music practices (vs. improvisation or novel sound generation)
- What is the rationale for display content?
- Is the interface usable and natural (for non-experts)?
- Is there a sense of involvement/immersion? (striking right balance with skill problem elimination)
- What is the effect of multimodal display, if any

Hypothesis (provided the skill problem is not an issue)

**Key Elements in enhancing the musical experience**

- Sensory Immersion
  - Visual field of view / 3D sound
  - First person viewpoint
  - Multimodality*
- Spatial and Logical Immediacy
  - Control range
  - Consistency in the display
  - (Performance feedback)
- Control Immediacy
  - Convincing metaphor (minimal cognitive load)
  - Minimum latency and synchronization

The 5th Glove for global tempo control

Vibration based tapping sensor
"Musical Galaxy" (circa 98): Demo

- Size of Earth: Present Tempo
- Stepping Stones: Notes
- Positions of Stepping Stones: Pitch
- Distance btw Stones: Duration
- Yellow: Past Notes
- Red: Present Note
- Blue: Future Notes

"Road of Tension" (circa 99)
Fakeplay (PC Version), 2001

Musical Submarine, 2003 (Exhibited at Korea Science Festival)

Perception of classical music …
But perhaps, at least in the virtual world …

Q: Christopher Hogwood, Daniel Barenboim, and Neville Mariner are all on the same plane when it ditches in the middle of the Atlantic Ocean. Who is saved?

A: Mozart