Matching Parts: Inner Voice Led Control for Symbolic and Audio Accompaniment

Nick Collins
Department of Informatics
University of Sussex
Falmer, Brighton, BN1 9QH
N.Collins@sussex.ac.uk

ABSTRACT

In sympathy with the oft-overlooked inner parts of music, this paper seeks the live control of outer parts by a matching process from the inner voices. This artistic conceit is instantiated in two prototypes; first, in a symbolic MIDI system where a MIDI pianist can control the playback of MIDI files truly from 'within', and second, via live concatenative synthesis constrained to apply audio signal matches to an inner voice filter range, controlling the playback of all remaining spectral parts. These techniques are presented more generally within a context of 'indirect' feature matching via a proxy part linked to a database of playback material. Representational issues and future extensions are discussed.

Keywords

Inner Parts, Feature Matching, Concatenative Sound Synthesis, Melodic Similarity, Accompaniment, Interactive Music System

1. INTRODUCTION

In much music, the perceptual status of the inner parts is somewhat de-emphasised, a situation explicable on psychoacoustic grounds. There is reason to believe that we attend to outer parts (an upper melody part and bass line, for example) first, and in many cases may even fill in the inner parts by a process of harmonic assumption and general textural listening, rather than fully resolving them [8, 4, 2]. Particularly in more homophonic circumstances it is difficult to track inner lines, unless very familiar with the parts (perhaps abetted by a score). More clearly elucidated polyphonic music might avoid common onsets between parts and employ careful vertical spacing to assist stream segregation. As auditory events, inner parts are most at the mercy of vertical integration processes and are less likely to form independent horizontal streams - thus contributing to auditory chimerae, in Bregman's terminology [8]. Nevertheless, those musicians accustomed to play such parts amongst timbrally homogenous streams, from viola and second violin players to altos and tenors, have a clearly important role (even if occasionally in terms of texture and harmony over independently perceptible part-writing).

The work in this paper reflects a hybrid relation to both accompaniment systems, and to audio cross-synthesis from feature matching into large databases, the latter area dubbed concatenative sound synthesis [9]. Accompaniment systems typically synchronise a lead melody to an accompaniment backing [7, 9]; they might form interesting options for our purposes if they were to align playback referenced from an inner part. However, this paper treats the circumstance of general audio and symbolic matching, allowing for approximate matches; whilst a conventional accompaniment system might be heuristically constrained to assume onwards, near-continuous score playback positions, cued random access of a database potentially allows heightened interactive control.

Thus, I argue for an aesthetically novel take on accompaniment. The control input is matched to a database indirectly, via proxy representatives; these hidden proxy parts are not in the eventual playback but act as a pertinent tag to linked database material. The inner voices are seen as one interesting proxy for matching.

Such processes may have interesting applications in music education, both from the perspective of prospective players of inner parts, but also for less skilled musicians (no offence meant): since middle parts are often perceptually less critical, they may form an interesting pivot from which to conduct other lines without having the full demand of playing those lines. However, the primary emphasis of this study will be on novel improvisational performance opportunities empowered by this mode. It is an interesting conceit to imagine primarily conducting the inner voices, with the outer parts hanging off as subsidiary outcomes of control, hence *inner voice led.*

Whilst describing the technology in sufficient detail in this paper to outline the compositional applications, the reader may wish to consult the references for further technical information on concatenative synthesis and feature matching [9], realtime interaction [2] and symbolic matching and melodic similiarity [5, 6, 3, 10].

2. SYMBOLIC MATCHING OF PARTS

Prototyping was carried out utilising that stalwart of the literature, MIDI piano music. In this first method, the playing of a MIDI piano provided the control mechanism. The performer only needs to play a single line, such as an inner part or specific monophonic track, and a matching procedure tries to find appropriate ensemble material.

There are two main stages; the preparation of a database from a MIDI file source, and the live matching procedure for referencing into that database. The choice of representation is critical, and the particular function of similarity between material is an important compositional decision [5, 3].

In a preprocessing stage the MIDI File was converted into a more suitable representation for manipulation, converting MIDI off events to durations and preparing inter-onset-intervals alongside absolute starting times. This facility was developed by the author as the MIDIFileAnalyse class, extending an existing SuperCollider MIDIFile read and write functionality (`http://www.informatics.sussex.ac.uk/users/nc81/courses/cm2/workshop.html`). The interactive system was also built in SuperCollider.

2.1 Matching note lists

An essential facet of this study is the choice of representation and algorithm for the determination of similarity. A window of recently played notes collected from the performer must be matched to those in the database; a two second window size was taken as the standard. To prepare the database, two second windows were taken every 0.1 seconds.

A 'note-wise' matching technique was implemented with a two pass algorithm described below; more complicated measures of distance, such as the Earth Mover's Distance and derivatives [10] were not instantiated for efficiency reasons in the live domain, though they may form the basis of future extensions. A point based rather than timeslice algorithm[1] was used to more accurately reflect the MIDI notes performed, avoiding strict boundaries, though for the latter histogramming could have been a little more efficient (with precalculation say of 100 millisecond time slices over 2 second segments for the whole database) and avoids finessing different length sequences in match score calculations.

The match procedure was carried out with respect to absolute time, rather than a beat based representation; whilst beat tracking would have been plausible (and indeed, there may be a benefit to further analysis in the pitch domain, such as key tracking), this version concentrated on the interaction possibilities of basic technique, and richer representations are reserved for future work.

MIDI notes in each window were transformed to normalised (proportional start time, proportional midi note pitch) pairs, which can be considered as two dimensional points in the range [0,1] by [0,1] [10]. If the two sets to be matched were A and B, the match scoring operated first to attach points in A to points with close start times in B, and then to look at Euclidean distances in (pitch,time) between these points. Pseudo code follows:

- without loss of generality, card(A) < card(B)
- (iterate forwards) for each a in A:
- Find the greatest point b in B not yet used such that time(b)<=time(a) , or record nil if none meets the criteria.
- (iterate backwards) for each a in A in reverse order
- Check for a closer b in B such that b>a and b is closer than any existing match for a, and such that b is not otherwise already assigned. Else assign nil.
- Now look at Euclidean distances between points, and impose a fixed penalty of 1 unit for any unmatched points (nil) in A and B.

[1]Hu et al. [6] call these options *event-based* and *frame-based.*

The best match is that segment in the database with minimum match score to the live control data window.

2.2 Pseudo code breakdown of the algorithm

Pseudo code is presented here for the database preparation stage:

For each short-term window store three elements:

1. The match representation of this fragment (this may be a proxy in terms of the desired inner voice range)
2. An index into the original MIDI File corresponding to the next event AFTER this window
3. The scheduling delay to the next event which should be played back after the match, i.e. the first AFTER this window

and for the interaction/playback stage, on receiving a new MIDI note event:

- Update the list of events over the last two seconds
- If a new match is allowed
    1. Form a target representative A from the current window
    2. For each member B of the source database (linear search) calculate the match score d(A, B) using the procedure in section 2.1; store the running minima of d
    3. From the minimal match, schedule playback of the appropriate outer parts (without the inner parts)

Various conditions for new match calculations to take place were posed; obvious possibilities were seeking a new match every N events (N=10 provided one compromise) or every x seconds (or beats if in combination with beat tracking). This decision embodies a tradeoff. It is also straight forward to determine the quality (closeness) of matching to the database and only allow a substitution if a numerical condition is met.

2.3 An Example Application

In a demonstration of inner part matching into a database, the F minor prelude by J.S. Bach (no. 36 of the 48) formed the material. The notes between MIDI note 55 and 69 were used for matching; segments were formed consisting only of notes in this range, as representatives for the other notes. live playing of notes in this same range would be matched to select playback material from a database of 1378 windows. The player had a sense of accessing Bach's piece from within, with scheduling smoothly controlled from the mechanisms outlined above.

3. AUDIO MATCHING OF PARTS

Audio matching shows similar processes to symbolic matching, except that the representation used for matching is a feature vector derived from the audio signal.

3.1 Concatenative Synthesis on Live Streams

The Concat UGen[2] for SuperCollider was designed for realtime use, allowing the concatenative cross-synthesis of two live streams [2], rather than working from a fixed pre-existing source database [9]. In operation, input signals are windowed in 256 sample frames, within which four features are extracted, two time domain (zero crossing rate and log power) and two frequency domain (spectral centroid and spectral rolloff, via a 256 point FFT). The features are normalised to the range 0-1 for sensible comparability of dimensions, via empirically derived transforms determined by observing feature values on a range of input signals. The feature vector for one input can be compared to a database of feature vectors captured over time for the other, seeking the best match according to a Euclidean metric.

The UGen was modified for these investigations to have a longer memory, with a time dependence in the feature vector distance metric via a weight vector which privileged the most recent frames [9, p.152]. This greater context assisted with the quality of matches delivered. A further modification was to add a third stream input; this was intended to act as the feed to the audio database, which would be indirectly accessed via the corresponding second stream analysed for features.

A match is sought if a certain user-specified time has passed since the last match operation, and if the input control stream amplitude is sufficient to trigger the process. The three input streams might be labelled the *control*, *proxy* and *store* streams, where:

control forms the momentary target for a match search (and the last N frames of its features are kept in memory).

proxy this audio stream is analysed for features and these are recorded in a (large, typically multiminute) buffer for later search. Such features are actually used as the hidden link to index the

store which is simply the actual sampled data store for playback from the database; the audio content can be different from the proxy, but will typically be correlated with it (perhaps as a different track of the same recording, or as the outer parts associated with the inner parts used for feature matching)

The search for the best match is carried out with respect to the following equation, which seeks the minimum scoring weighted Euclidean metric:

$$\min_k \sum_{j=0}^{N-1} w_j \sum_{i=1}^{M} (x_j(i) - y_{k-j}(i))^2 \qquad (1)$$

where N is the number of frames of memory, and M is the number of features (N=10 and M=4 were the defaults), k ranges over all pertinent frames in the database, and $x_j(i)$ is the ith feature of the jth frame of signal x. x represents the control input feature vector, y the proxy representative feature vector, and w a weight vector which favours lower j corresponding to more recent times (various weight vectors based on a geometric falloff were empirically tested).

[2]http://www.informatics.sussex.ac.uk/users/nc81/code

Rather than taking a straight Euclidean distance[3], the weighting was introduced to give some sense of time dependence. Without it, there is no need for a matched frame from the database to have the feature values in the same ordering as the target. There are inherent problems in the reduction from a multidimensional feature vector (implicitly ordered in time, with M*N dimensions) to a single value.

Because the UGen works on a circular buffer rather than a fixed and pre-existing buffer database, it was not possible to take advantage of certain possible speed-ups. The architecture of the plug-in necessitated a linear search, whereas a fixed buffer would have allowed the precalculation of feature vectors and the use of much more efficient kD-tree or Ball-tree searches [11] or approximate nearest neighbour search [1] through the database in matching. However, this was not a major issue for prototyping herein (and one immediate efficiency trick is to exit any distance calculation as soon as it goes over the current best minimum), though could be tackled in future work, especially if proceeding to very large databases.

3.2 Applications

Given a multitrack recording (with relatively isolated parts), one artistic application of the technology just described is to use a single track to control the playback of the remainder (Figure 1). The virtue of this is to match to a similar timbre to the control instrument; for instance, the voice might be used to match to a vocal track or a wind instrument. This then indirectly leads to the co-option of the other tracks as a backing or abstract accompaniment to the primary control. The other tracks 'come along for free' in the matching procedure.

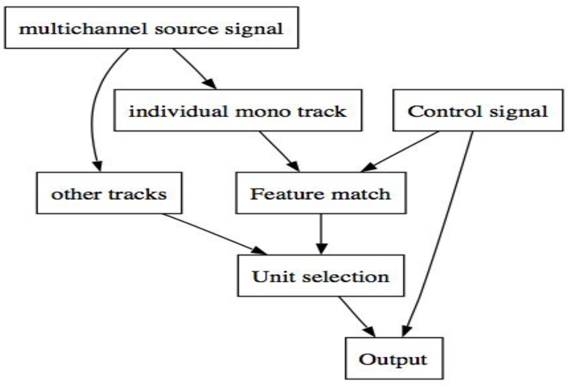

Figure 1: Feature matching of a control input to one track of a multitrack, determining playback of the remaining tracks

In a first experiment, the author used 'stereo' Beatles recordings; as is well known, these recordings provide interesting cases of channel mixing of greater cleanliness and separation than modern mixing techniques. I've always hated 'Yellow Submarine', and the opportunity to remix it via matching my own voice to Ringo's was too great a temptation to resist. Because in this recording the vocals are predominantly on the right channel, and the drums, guitar

[3]Without square root, this being unnecessary for finding the minima.

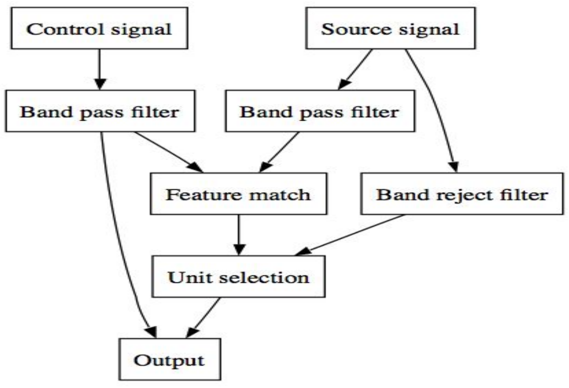

Figure 2: Filter based simulation of inner part matching (without complex transcription methodology), controlling playback of a composite of control signal inner and matched outer spectral content

and whooshes on the left, I could cue my own Beatles accompaniment indirectly via the vocal part and suppress Ringo at the same time.

In another example, an isolated microphone two track recording of a baroque recorder and harpsichord playing a sonata (already sourced for [2]) was navigated. A direct feature match of my voice to the (polyphonic) harpsichord gave much weaker results than going indirectly via the (monophonic) recorder part, this in turn selecting unit playback from the corresponding harpsichord accompaniment.

Melodic extraction of an inner part remains an unsolved problem; it is only reasonable for outer parts with current technology, and may remain inherently speculative and ill-defined in many situations due to those chimerical sound objects discussed in the introduction. To simulate feature matching from inner parts, complementary band pass and band reject filters were used in the signal processing setup detailed in Figure 2. A recording of Lassus' *Tenebrae factae sunt* from *6 Responsories pour l'office des Ténèbres du Vendredi Saint* for four voices provided one target – by singing in the range of a tenor or low alto I could cue outer parts around me, and the blend gave a feeling of playing somewhat 'within' the music analogous to the symbolic Bach prelude example.

To be fair, these audio effects were within the limitations of concatenative synthesis, which could lead to modulation effects with fast jumps of location for short granularities – longer playback windows were preferred. The point at which new matches should be triggered was not particularly controlled; I did not make it manual, and nor have I yet implemented more specific tracking processes for my own voice that might provide better integration. Nevertheless, the core idea was successfully prototyped.

4. CONCLUSIONS

The algorithms described in this paper could certainly be improved, but the priority in the context of live performance was to prototype the interactive possibilities. In this duty, the author feels that these techniques offer interesting potential for novel musical exploration, not just for expert musicians, but also potentially in music education especially as a way of 'exploring musical works from within'.

Whilst the signal processing and psychological issues of symbolic and audio representation remain open questions, and have an impact on the effectiveness of any matching procedures, the composer can add additional noise or modified weights in the matching equation (or indeed, rival representations and metrics) if alternative compositional aesthetics so demand. The match procedures described in this paper also sidestep the latency problems of immediate reaction, by seeking material which continues the current match window.

The symbolic matching section of this paper deals with a pre-prepared database and the audio matching with multiple live streams. These positions could be inverted, and live database formation for MIDI material might form an intriguing performance situation, whilst a pre-analysed database for audio would provide more efficient search possibilities, and much larger source material (particularly at the opening of a set). Feature matching from a multi track MIDI recording analogous to the multitrack audio could also be employed; an example might be to work with a MIDI file of a string quartet (or a much larger database of string quartets across history from Haydn to Ligeti), cueing the ensemble from the viola part, an accompaniment process subverted to novel explorations.There remains much fertile area to explore.

5. REFERENCES

[1] M. Casey and M. Slaney. Song intersection by approximate nearest neighbour search. In *Proc. Int. Symp. on Music Information Retrieval*, 2006.

[2] N. Collins. *Towards Autonomous Agents for Live Computer Music: Realtime Machine Listening and Interactive Music Systems.* PhD thesis, University of Cambridge, 2006.

[3] E. Gómez, A. Klapuri, and B. Meudic. Melody description and extraction in the context of music content processing. *Journal of New Music Research*, 32(1), 2003.

[4] S. W. Hainsworth. *Techniques for the Automated Analysis of Musical Audio.* PhD thesis, University of Cambridge, 2004.

[5] W. B. Hewitt and E. Selfridge-Field, editors. *Melodic Similarity: Concepts, Procedures and Applications (Computing in Musicology II).* MIT Press, Camb, MA, 1998.

[6] N. Hu, R. B. Dannenberg, and A. L. Lewis. A probabilistic model of melodic similarity. In *Proc. Int. Computer Music Conference*, 2002.

[7] C. Raphael. Synthesizing musical accompaniments with Bayesian belief networks. *Journal of New Music Research*, 30(1):59–67, 2001.

[8] E. D. Scheirer. Bregman's chimerae: Music perception as auditory scene analysis. In *Proc. Int. Conf. on Music Perception and Cognition*, 1996.

[9] D. Schwarz. *Data-driven Concatenative Sound Synthesis.* PhD thesis, Université Paris 6, 2004.

[10] R. Typke, P. Giannopoulos, R. C. Veltkamp, F. Wiering, and R. van Oostrum. Using transportation distances for measuring melodic similarity. In *Proc. Int. Symp. on Music Information Retrieval*, 2003.

[11] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques (2nd Ed).* Morgan Kaufmann Publishers, San Francisco, 2005.