Contrary Motion: An oppositional interactive music system

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

ABSTRACT

The hypothesis of this interaction research project is that it can be stimulating for experimental musicians to confront a system which 'opposes' their musical style. The 'contrary motion' of the title is the name of a MIDI-based realtime musical software agent which uses machine listening to establish the musical context, and thereby chooses its own responses to differentiate its position from that of its human interlocutant. To do this requires a deep consideration of the space of musical actions, so as to explicate what opposition should constitute, and machine listening technology (most prominently represented by new online beat and stream tracking algorithms) which gives an accurate measurement of player position so as to consistently avoid it. An initial pilot evaluation was undertaken, feeding back critical data to the developing design.

Keywords

contrary, beat tracking, stream analysis, musical agent

1. INTRODUCTION

Computational agents as musical interlocutants have been extensively studied [13, 3], though the degree to which they can comfortably demonstrate independent but appropriate musical action in real contexts reflects the deep challenges in machine listening technology and music generation. Many 'interactive' systems turn out to be directly reactive slaves, or even if exhibiting some autonomy, lag behind human auditory and cognitive capabilities. It can be productive to stimulate musicians through systems which use algorithms far flung from traditional musicianship (for example, the emergent swarm systems of Impett [8]). However, intersection with human musical practice remains the gold standard of progress, and avoids any claim that mappings from more mathematical spaces are always tempered by human auditory cognition. Is there another way we might stimulate new music making, while continuing to tackle core problems in the simulation of human musicianship? The project described in this paper approaches this problem by using machine listening technology as a frontend for a system which then 'pushes against' a human performer, opposing aspects of their decisions in the hope that this provides a fruitful and inspirational encounter.

A computer musical system entitled 'Contrary Motion' is a rich ground for a conference on new interfaces for musical expression, for aside from advances in machine listening for interactive systems and reflection on interaction, the new aesthetics of such a system is part of exploring the space of potential musical expression opened up by computer music. The opportunity here is for a system that continually prompts you outside of your comfort zone, a journey which is intended to be an inspiring one for an experimentally minded musician. Famously, John Cage was always ready to embrace music which he didn't like [10]; Frank Zappa describes listening and over to over to records of Varese to integrate their initial otherness into his aesthetic [16]. These examples show that opposition can be a temporary state on the path to new musical awareness, a trait also theorised in coverage of putative 'noise' musics [6, 14]. A more energetically diametrical response may also be of interest in some personal or even therapeutic settings, though overt confrontation is not to everyone's taste!

It should be admitted up front that the form of opposition is determined by the space of possible action, which is necessarily tempered by the system designer. The context of this paper is MIDI piano, a situation which none the less admits challenges in machine listening, and sufficient richness of repertoire and practice to be a strong basis for this study. Contrary Motion incorporates agent-based[1] beat tracking and stream segregation algorithms, analysis of pitch content over various timescales, and responses which exploit this online analysis, using it for various oppositional strategies. The system currently assumes MIDI piano input and output, thereby restricting pitch materials to the 88 note range from MIDI notes 21 to 108. The primary implementation platform was SuperCollider, and source code is available on request; for output, the system sends MIDI out messages to a sampler with ready to use high quality instrument sounds (provided by Logic MainStage). This paper first describes the machine listening algorithms, before the contrary musical response generation, and discussion and evaluation of the system in action.

2. MULTI-AGENT BEAT TRACKING

The beat tracking algorithm outlined here was developed to enable fast reactions in a dynamic performance setting. It is an IOI (Inter Onset Interval) based scheme which avoids histogramming in favour of a multitude of short-lived agents at particular periods and phases. One primary assumption is that beats are (occasionally) marked by ac-

[1] Agent is used here in the weak sense of an operating instance, such as a single active hypothesis or module, and not in the stronger sense of autonomous agency associated with a complete complex cognitive system

NIME2010 June 15-18, 2010, Sydney, Australia

tual onsets, and that evidence for beats will accrue from on-beat or binary subdivided off-beat hits. It is also assumed that wider cycles with non-isochronous subdivisions are not themselves critical in forming the metrical structure, so that a clear isochronous beat is in effect. Whilst extensions for triplets are feasible, the binary subdivision setting is sufficiently common in musical practice to cover a large enough space of action, and evidence for beats from triplets can still accrue on beat rather than from beat subdivisions. Precedents for the algorithm might be found in work by Dixon with his agent-based conception of active hypotheses [5], in the capture fields of nonlinear oscillator models [11] (though without reducing capture window over time), and in the IOI quantisation and particularly the conceptualisation of an expectancy field of Desain and Honing [4].

2.1 Finding a winning period and phase

Algorithm 1 Beat tracking algorithm pseudocode

Input: New onset time *now*, list of active agents, each with its own period, phase and running score, and a list of active times, being recent onset times within the last second

Output: A new winning period and phase, updated lists of active agents and times

if time since last onset < 0.04 **then**
    weight down this new onset as just an asychronous chord member
end if
for each active agent **do**
    update score for agent based on how well it predicts *now* (see text)
end for
cull any agent whose score is less than -3.0 and which has not been a winner in the last four seconds
Find agent with current max score; this gives the best period and phase estimate
cull any active times which are not within one second of *now*
for each active time *then* **do**
    $ioi \leftarrow now - then$
    if $ioi > 0.25$ **then**
        add to the list of active agents a new agent initialised with phase *now*, period *ioi* and score 0
    end if
end for

Algorithm 1 gives pseudocode for the algorithm, for the central step of updating the beat tracker when a new onset is detected (for MIDI, a new MIDI note on event occurs). A list of active agents is maintained, where each agent has its own period and phase, and an associated score to track how well it has been predicting the timing of events. Multiple agents can have the same period and phase, but will differ in their scores based on their histories. All agents are derived from some IOI between observed time points closer than a second in time (not necessarily consecutive events, but separated by no more than a second, corresponding to 60bpm). An Inter-Beat Interval (IBI) derived from an IOI must be at least 0.25 seconds long, following London's observations [12] (corresponding to 240bpm).

To avoid bias from chords, any onsets closer than 40 milliseconds are weighted down in their effect on scores (multiplied by the empirically determined value 0.5 for each consecutive close note), though they can still contribute to the seeding of new agents.

Unsuccessful agents are not expected to live long, and this fast turnover of many agents (around 20-100 were active at one time in tests) is central to reactivity. New IOIs of sufficient size are always immediately available as potential IBIs, and if reinforced (or if nothing is scoring well) can very quickly take over. A beat tracker constructed in this way is more adaptable than a standard cross-correlation or autocorrelation on a two to six second window. Scoring is critical; some tolerance is required in matching an agent's predictions to new observed data, in the face of expressive performance timings. A match is scored as an on-beat within 30 milliseconds each side of a predicted location, scoring the standard weight of 1.0. Off-beat matches at eighth note locations are scored at half value. Otherwise, missing a given onset time gives a penalty of the note's weight. This means that an agent must keep predicting a certain proportion of events if it is to have any longevity; it otherwise gets a larger negative score and is killed off.

Refinements to this basic set-up would involve scoring triplets or other unequal subdivisions, and imposing a resonance curve for preferred tempi. The former adds complexity, in that the more possibilities there are, the more accidental coincidences occur, keeping some agents alive unnecessarily. The latter was not found important in practice at this juncture. Weights can also be made velocity dependent, so that accents contribute more to match scores and missed note penalties.

2.2 Finding an opposing metrical structure

A second stage in rhythmic analysis for this system deals with the contrariness. Here, a maximally dissimilar period and phase is desired for the computer from those tracking the human participant. An expectancy field is formed using all active beat agents of sufficient trustworthiness (scores over some threshold, for example 0.0). Two histograms are formed, one for periods (i.e., reciprocals of tempi), and one for predicted beat locations. These histograms are then searched for a region of maximal sparsity, being a run of consecutive zero scores of at least five bins in a row, or failing that the minimal total score within a ten bin local window stepped through the histogram. The bin central to the zero run or at the centre point of the window is then selected.

For a dissimilar period, a histogram is first constructed from all the current agent beat periods. The mapping $((period - 0.25)/0.75)^{0.63092975357146}$ is used to give more histogram bins for smaller periods (higher tempi), so as to reduce a bias to larger periods implicit in linear rather than logarithmic treatment of tempi. This mapping gives a normalized number between 0.0 and 1.0 which can then be scaled within 100 histogram bins by $\lfloor number * 99.999 \rfloor$.

Next, to determine a location where the human is likely not to be, a histogram is created of possible beat locations over the next second, with a resolution of 20 milliseconds (so, 50 histogram bins). For each agent, the appropriate histogram bin is incremented for any following locations of beats predicted by a given agent (there may be more than one in the next second depending on the agent's tempo), weighted by that agent's current score. This initial histogram can be further modulated by the already chosen period; phases are searched through only up to the period size, and the weightings increased at particular corresponding bins up to two periods later (depending on the period in the first place and the fit within one second). This emphasizes that the minimally expected phase must avoid multiple expected beat locations in the future.

Figure 1 illustrates this situation. The winning period and phase form the basis for scheduling the next second worth of events. As well as assessment of data, the algo-

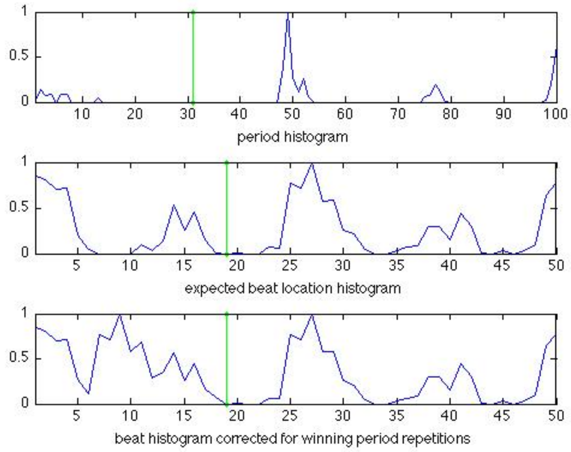

Figure 1: Example histograms for finding opposing metrical structure (from a real test case): the top shows the periodogram of observed periodicities. The expectancy field of beat locations is the middle diagram, and the bottom shows the corrected histogram taking into account the selected period. Solutions are shown by the vertical lines.

rithm was tested live empirically by sonifying the predicted next beat according to the beat tracker, and at a different pitch, a sequence of beats illustrating the chosen oppositional period and phase.

3. COMPARISON OF PITCH STRUCTURE AT MULTIPLE TIMESCALES

The system observes pitch content at a number of timescales, and then compares analysis windows both for changes within a given scale, and changes between different scales, to build up a picture from highly local to more global change. The pitch content itself consists of histograms over the 88 piano notes. Whilst pitch classes could provide a further viewpoint, chroma equivalence classes are not currently used for their loss of registral and directional information. Histogram contributions for notes can be weighted by velocities. One form of inversion for the pitch histogram as a distribution is to find the maximal value M across the bins, and form a new histogram with values $M - oldbinvalue$ for each bin, normalizing the probability mass to one.[2]

Differences are scored as the absolute difference of histogram vectors, summed.The timescales involved are windowsizes of 0.25, 0.5, 1, 2, and 4 seconds, at hops of 0.25, 0.25, 0.5, 1 and 1 seconds respectively. Corresponding to these five levels, the number of previous analyses stored is 8, 8, 4, 2 and 2, in total covering regions of 2, 2.25, 2.5, 3 and 5 seconds respectively. Comparison of pitch content at the same time scale looks at changes between successive windows of the same size, reflecting stability over a particular time scale; total change can be summed over the sequence of windows, for a single measure of 'choppiness' in pitch content. A between-scale comparison aligns windows from consecutive scales, summing differences after compensating the smaller scale histogram for the difference in window size (all entries are multiplied by this scale factor). The assumption here is that if pitch content continued along the same lines as a given window, with respect to the larger scale's window size, what would we expect to see? This measures how effective the smaller scale is as a prediction of larger scale content (the smaller the difference, the better the prediction was), and hence how more local structure compares to larger-scale structure.

The comparative information on timescales is currently used to make a single function of the between scale data, indicating a local or global bent. In particular, the focus is on changes between the one second pitch histogram, and the two second, normalizing by the globally observed minima and maxima so far. Lower values are indicative of greater pitch content stability, which can be opposed by greater variation in response. The full potential of such measurements, however, has not been fully exploited in musical response generation in the project so far.

[2]The operation could be expressed in SuperCollider code as: (maxItem(histogram).dup(88)- histogram).normalizeSum.

4. MULTI-AGENT STREAM TRACKING

Algorithm 2 Stream tracking algorithm pseudocode

Input: Time *now*, new *chord* of notes (each with associated *notetime* and *notepitch*), and a list of active streams, each with its own current score and list of notes (including the *lasttime* and *lastpitch* a note was assigned)

Output: Updated lists of active streams, predominant stream as highest scoring

for each active stream **do**
    Set a flag for this stream to true (available to collect notes)
end for
$unassignednotes \leftarrow$ empty list
for each *note* in *chord* **do**
    $winner \leftarrow$ nil
    $mindistance \leftarrow 99999.9$
    for each available (flag is true) active *stream* **do**
        $timeseparation = notetime - lasttime$
        $pitchseparation = \|notepitch - lastpitch\|$
        if $timeseparation < 1.0$ AND $pitchseparation < 12$ **then**
            $distance = timeseparation^2 + (pitchseparation/12)^2$
            if $distance < mindistance$ **then**
                $winner \leftarrow stream$
            end if
        end if
    end for
    if $winner \neq$ nil **then**
        Extend winning stream by *note*, and mark its availability flag to false; increase stream's score by $2 - distance$
    else
        Add *note* to *unassignednotes*
    end if
end for
Cull any active streams which have not been assigned a new note within one second of *now*
Find stream with current max score; this is the most active stream (that which explains the most notes in the most parsimonious way)
Initialise a new active stream for every *note* in *unassignednotes*

A performance robust online stream segregation algorithm was created, to provide further insight into human playing. It is not a perfect solution to the difficult problem of tracking multiple simultaneous voices within a musical context [2, 17], but a pragmatic attempt to give the system a greater sensitivity. It uses no further musical knowledge than a notion of general proximity in time and pitch.[3] Subtleties of the stream segregation problem, such as the tendency to connect notes at wider pitch leaps if spaced further apart in time (as investigated by van Noorden), are not accounted for. Typical existing algorithms might be based on dynamic programming, statistical models or lists of heuristics, and the problem is usually investigated in offline situations rather than for live systems [9]. For a practical realtime solution, a similar coding structure is exploited to the beat tracking algorithm, creating an agent based system where each agent is one currently active stream.

The algorithm is intended to update its state given one arriving note at once. A subtlety is that chords often arrive with closely spaced notes, that could cause confusion (by proximity in time, each note of the chord may enter a single stream, rather than being available to different active streams). This can be coped with by delaying judgement until there is a sufficient inter onset interval to indicate a completed chord, before assigning each chord note to one stream (the interval is at least 0.04 seconds; this necessitates running a process in the background to check if more than 0.04 seconds has passed since the arrival of a last chord note, in the absence of newly arriving notes). Algorithm 2 details the procedure in updating active streams and assignments.

As per the beat tracking algorithm, unsuccessful streams are not intended to live long. However, the streams always represent active voices, rather than potential hypotheses, so a multiplicity of solutions are not maintained, at the expense of missing some longer term phenomena in stream crossing, but at a gain in simplicity of implementation.

Streams are useful to explore generalised contrary motion effects. A stream detected in the player's input, which goes up, might influence a counter-riposte where the computer heads down. In general, envelope shapes for pitch content selection can be extracted; the region of pitch histogram availability might follow this shape, or an inverse of it. All the detected streams may form the basis of reaction, or just one; a primary stream is tracked as the highest scoring (usually that which is longest lived and/or most compact in pitch and time).

5. RESPONSE GENERATION

Whilst I have experimented with a number of timescales, and the beat and stream analysis is updated with or within 0.05 seconds of every newly arrived note, in the standard system new large-scale response decisions are made once per second, taking into account the previous one to three seconds. This timescale of action corresponds well to human working memory and the perceptual present [1].

For rhythmic materials, the beat tracker provides a quantisation solution. Assuming as a simplification that only 16th notes are dealt with, recent onsets can be interpreted with respect to the winning period and phase hypothesis. There are 16 possible patterns within one beat (2^4, for a binary choice of rest or note at four positions). All *full* beats within the last three seconds, according to the beat phase, can be analyzed for their rhythmic pattern. A histogram of frequencies of occurrence of each pattern can be created, and then inverted to create an oppositional distribution; the default in the case of an empty event list is a uniform distribution. The inverted distribution is then used to select patterns for the computer to play back, as many as fit in the available beats.

Although human silence might have been met with computer maximum activity, this seemed a step too far; deliberate opposition on every facet may over stretch, and 'Wessel's rule of thumb'[4] was respected. The system should shut up when the human performer does, at least within some bounds of interaction. So the number of notes in play was determined from the number detected, typically the number in the previous one second time window.

A number of playing modes were created; the two main ones evaluated used the stream analysis, and were:

1. From the primary stream, extract a list of pitch intervals and invert them. Select a starting note by analysing the pitch histogram for a less likely position (as per the beat histogram analysis in section 2.2). Generate IOIs using the rhythmic generation process detailed above, for as many actual notes as desired.

2. For each active stream, find the content from the last X seconds; generate a response stream (over the next Y seconds) which inverts the pitch intervals, and scales the time intervals by the ratio of the detected tempo to the oppositional tempo. (X and Y were both 1 in a basic implementation).

I also experimented with a mode zero, where the inverted distribution for pitches was used for independent draws of output notes (rather than a strict musical inversion operator), the IOIs were sourced from the rhythm model as above, and if there were more notes to play than room in the rhythm, chords were generated by assigning extra notes in modulo fashion.

6. MUSICAL EVALUATION

The author, an experienced pianist, had played with the system constantly during its preparation and fine tuning, for direct input to the design cycle. Yet, in order to assess outside impact, the system must be made available to fresh ears and hands. A small pilot study let other musicians play with the system, who had never before encountered it. Three musicians took part in this task; none were advanced pianists, but all had practical experience of keyboard playing in accompaniment, experimentation and performance tasks. This was even preferable, since sheer virtuosity is not pre-requisite of a musical encounter with the system.

For evaluation sessions, which each lasted around half an hour,[5] the human part was synthesized with a piano sound, and the computer part with a marimba, so as to clearly differentiate contributions. The basic experimental ordering followed standard HCI practice [18, 15, 7]; musicians were first given a try-out without any knowledge about the system, for the two main playing modes (lead stream reaction, and all streams inverted). They then shared their impressions and thoughts. The experimenter then revealed the greater context of the project. The musicians were given a further chance to try the system in the two modes. And again, they had chance to reflect on the interaction.

A sample of comments are now summarised. The overall general impression was one of more atonal music than tonal music common practice; the pitch histogram inversion works within the chromatic aggregate, not within a diatonic or other limited scale set, and the rhythmic opposition builds in multiple metres. The most requested new feature was a richer musical knowledge of harmony, respecting for instance when a player actually performed within a minor

[3]For piano music, one might further analyse the position of hands and fingers as a physical constraint on voicings.

[4]As outlined at a panel on interaction at ICMC2005 in Barcelona; also see [19]

[5]They all overran, which showed that there was plenty to discuss!

scale; perhaps oppositions could be more locally reactive, working within recent pitch materials only, or recognising certain scale sets or thematic spaces. The pitch histogram inversions in this case proved just a first step. The contrary motion properties of Contrary Motion in the second playing mode in particular were quickly discovered by participants. In critiquing the system, a worry was expressed that some responses were too reactive, or hard to comprehend as organised; further tests might compare a stripped down version of the system based on uniform distributions for pitch and time with the full system itself. However, when playing with the system in a continuous setting, all participants appreciated the system's attempts to avoid their pitch and time locations.

A surprising finding for the author was that the system proved more deterministic than expected, exhibited when the musicians deliberately tested out repeating figures to assess the extent of variation (their exploration strategies differed very much from the author, taking in sparse playing, and in one case an emphasis on call and response where the system designer had anticipated much more continual joint performance as the standard). Whilst there are probabilistic elements in responses, under certain conditions the beat and stream segregation algorithms give similar answers, and the pitch choices particularly in playing mode 2 follow a linear map, exhibiting predictable behaviour.

As the experimenter had hoped, participants did attempt to catch the system out (especially in mode one or zero where the anti-beat tracking has the most obvious role), playing a musical game of syncing or not syncing: 'I was trying to catch it'. Their attitudes and tactics changed in this sort of way (understandably so!) when playing with the system after the explanation of its premises. One musician brought up parallels to techniques used by some breakcore artists (associated for example with the Wrong Music label) to frustrate dancing crowds expecting the beat, and indicated an interest in a drum or breakbeat based version of the system. They also suggested applications in training percussionists.

7. CONCLUSIONS

Contrary Motion is by no means a finished system, but enough has been built to assess it as a proof of concept. Richer third party feedback has been solicited to influence the design process outside of the immediate author. The oppositional stance can potentially provide a rich stimulant to musicians, though most likely of primary interest in experimental music, or those seeking something to refresh their palettes. To make an oppositional system requires consideration of the space within which any counter-gesture has meaning. This situation also presents a novel challenge for machine listening, in that to do this properly for a live interaction system requires a command of the human performer's position, and this project provided an interesting context for work on online beat tracking and stream segregation.

Future directions for the system would involve greater chord knowledge and other tonal/atonal theory, and more extensive machine learning facilities, so as to redefine the action space in and between performances. Time series analysis is not used to any great degree at present, and the prior or online construction of a database of musical materials with respect to which 'maximal dissimilarity' can be explored may involve porting work from music information retrieval. Further technical and interaction evaluations would be carried out for revised systems.

8. ACKNOWLEDGMENTS

Many thanks to the musicians who tried out this system. Furtherless, though I am indebted to all the anonymous reviewers of this paper, I wish to particularly single out 'Reviewer 2', whose opposition to this work proved most fruitful in reflection.

9. REFERENCES

[1] J. R. Anderson. *Learning and Memory: An Integrated Approach (2nd edition)*. Wiley, New York, NY, 2000.

[2] A. S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. MIT Press, Camb, MA, 1990.

[3] N. Collins. Musical robots and listening machines. In N. Collins and J. d'Escrivan, editors, *Cambridge Companion to Electronic Music*, pages 171–84. Cambridge University Press, Cambridge, 2007.

[4] P. Desain and H. Honing. *Music, Mind and Machine: Studies in Computer Music, Music Cognition and Artificial Intelligence*. Thesis Publishers, Amsterdam, 1992.

[5] S. Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 30(1):39–58, 2001.

[6] P. Hegarty. *Noise/Music: A History*. Continuum, New York, NY, 2007.

[7] W. Hsu and M. Sosnick. Evaluating interactive music systems: An HCI approach. In *Proceedings of New Interfaces for Musical Expression (NIME)*, 2009.

[8] J. Impett. *Computational Models for Interactive Composition/Performance Systems*. PhD thesis, University of Cambridge, 2001.

[9] A. Jordanous. Voice separation in polyphonic music: A data-driven approach. In *Proceedings of the International Computer Music Conference (ICMC)*, Belfast, 2008.

[10] R. Kostelanetz. *Conversations with Cage*. Limelight Editions, New York, NY, 1988.

[11] E. W. Large and M. R. Jones. The dynamics of attending: How people track time-varying events. *Psychological Review*, 106(1):119–59, 1999.

[12] J. London. *Hearing in Time: Psychological Aspects of Musical Meter*. Oxford University Press, New York, 2004.

[13] R. Rowe. *Machine Musicianship*. MIT Press, Cambs, MA, 2001.

[14] L. M. Schafer. *The Thinking Ear*. Arcana Editions, Toronto, 1986.

[15] A. Sears and J. A. Jacko, editors. *The Human-Computer Interaction Handbook (2nd Edition)*. Lawrence Erlbaum Associates, New York, NY, 2008.

[16] J. Slaven. *Electric Don Quixote: The Definitive Story Of Frank Zappa*. Omnibus Press, London, 1996.

[17] D. Temperley. *The Cognition of Basic Musical Structures*. MIT Press, Camb, MA, 2001.

[18] M. M. Wanderley and N. Orio. Evaluation of input devices for musical expression: Borrowing tools from HCI. *Computer Music Journal*, 26(3):62–76, Fall 2002.

[19] D. Wessel and M. Wright. Problems and prospects for intimate musical control of computers. *Computer Music Journal*, 26(3):11–22, 2002.