STRATEGIES FOR MANAGING TIMBRE AND INTERACTION IN AUTOMATIC IMPROVISATION SYSTEMS

Bill Hsu
San Francisco State University
18 PUBLICATIONS **123** CITATIONS
SEE PROFILE

Strategies for Managing Timbre and Interaction in Automatic Improvisation Systems

William Hsu

ABSTRACT

Earlier interactive improvisation systems have mostly worked with note-level musical events such as pitch, loudness and duration. Timbre is an integral component of the musical language of many improvisers; some recent systems use timbral information in a variety of ways to enhance interactivity. This article describes the timbre-aware ARHS improvisation system, designed in collaboration with saxophonist John Butcher, in the context of recent improvisation systems that incorporate timbral information. Common practices in audio feature extraction, performance state characterization and management, response synthesis and control of improvising agents are summarized and compared.

Timbre is an important structural element in free improvisation. When working with a saxophonist or other instrumentalist whose practice incorporates extended techniques and timbre variations, it is important to go beyond note-level MIDI-like events when constructing useful descriptions of a performance. For example, a long tone may be held on a saxophone, with fairly stable pitch and loudness, but the intensity of multiphonics is increased through embouchure control. An experienced human improviser would perceive and respond to this variation.

In this paper, I will focus on systems designed to improvise with human musicians who work with information beyond note-level events. I am mostly interested in what Rowe calls *player paradigm systems* [1], that is, systems intended to act as an artificial instrumentalist in performance, sometimes dubbed *machine improvisation* systems. Blackwell and Young [2] also discuss the similar notion of a *Live Algorithm*, essentially a machine improviser that is largely autonomous.

Earlier machine improvisers typically use a pitch-to-MIDI converter to extract a stream of MIDI note events, which is then analyzed; the system generates MIDI output in response. The best known of these systems is probably George Lewis's Voyager [3]. With the increase in affordable computing power, the availability of timbral analysis tools such as Jehan's analyzer~ object [4] and IRCAM's Zsa descriptors [5], and related developments in the Music Information Retrieval community, researchers have built systems that work with a variety of real-time audio features.

Since 2002, I have worked on a series of machine improvisation systems, primarily in collaboration with saxophonist John Butcher [6,7]. Butcher is well known for his innovative work with saxophone timbre [8]; all my systems incorporate aspects of timbral analysis and management. Our primary design goals were to build systems that will perform in a free improvisation context and are responsive to timbral and other variations in the saxophone sound. The latest is the Adaptive Real-time Hierarchical Self-monitoring (ARHS) system, which I will focus on in this paper. In addition to Butcher, my systems have performed with saxophonists Evan Parker and James Fei and percussionist Sean Meehan.

Butcher is mostly interested in achieving useful musical results in performance, rather than exploring aspects of machine learning or other technologies. To this end, I have tried to identify simple behavioral mechanics that one might consider desirable in a human improviser and emulate them with some straightforward strategies. One novel aspect of my work is the integral and dynamic part that timbre plays in sensing, analysis, material generation and interaction decision-making.

Collins [9] observed that a machine improviser is often "tied . . . to specific and contrasting musical styles and situations." For example, a system that tracks timbral changes over continuous gestures may be a poor match for an improviser playing a conventional drumkit. We are attracted to the idea of a *general* machine improvisation system that can perform with *any* human instrumentalist. This generality is perhaps mostly appropriate as an ideal for the decision logic of the Processing stage, detailed below. Specific performance situations inform our design decisions, especially in the information we choose to capture and how we shape the audio output.

Related Systems

In this survey of machine improvisation systems that work with timbral information, I will identify the goals and general operational environments for each; discussion of technical details will be postponed to subsequent sections. This is not intended as an exhaustive survey; however, the projects overviewed do represent a spectrum of the technologies used in such systems.

An earlier system is Ciufo's *Eighth Nerve* [10], for guitar and electronics, which combines sensors and audio analysis. Processing and synthesis are controlled by timbral parameters. His later work, *Beginner's Mind* [11], accommodates a variety of input sound sources and incorporates timbral analysis over entire gestures.

Collins [12] describes an improvisation simulation for human guitarist and four artificial performers. His emphasis is on extracting event onset, pitch and other note-level information from the input audio.

Yee-King's system [13] uses genetic algorithms to guide improvisations with a saxophonist and a drummer. The improvising agents use either additive or FM synthesis.

Van Nort's system [14] works with timbral and textural information, and performs recognition of sonic gestures in performance. It has been used primarily in a trio with accordion and saxophone.

William Hsu (musician, educator), San Francisco State University, 1600 Holloway Avenue, San Francisco, CA 94132, U.S.A. E-mail: <whsu@sfsu.edu>.

See <userwww.sfsu.edu/~whsu/improvsurvey.html> for audio documentation related to this article.

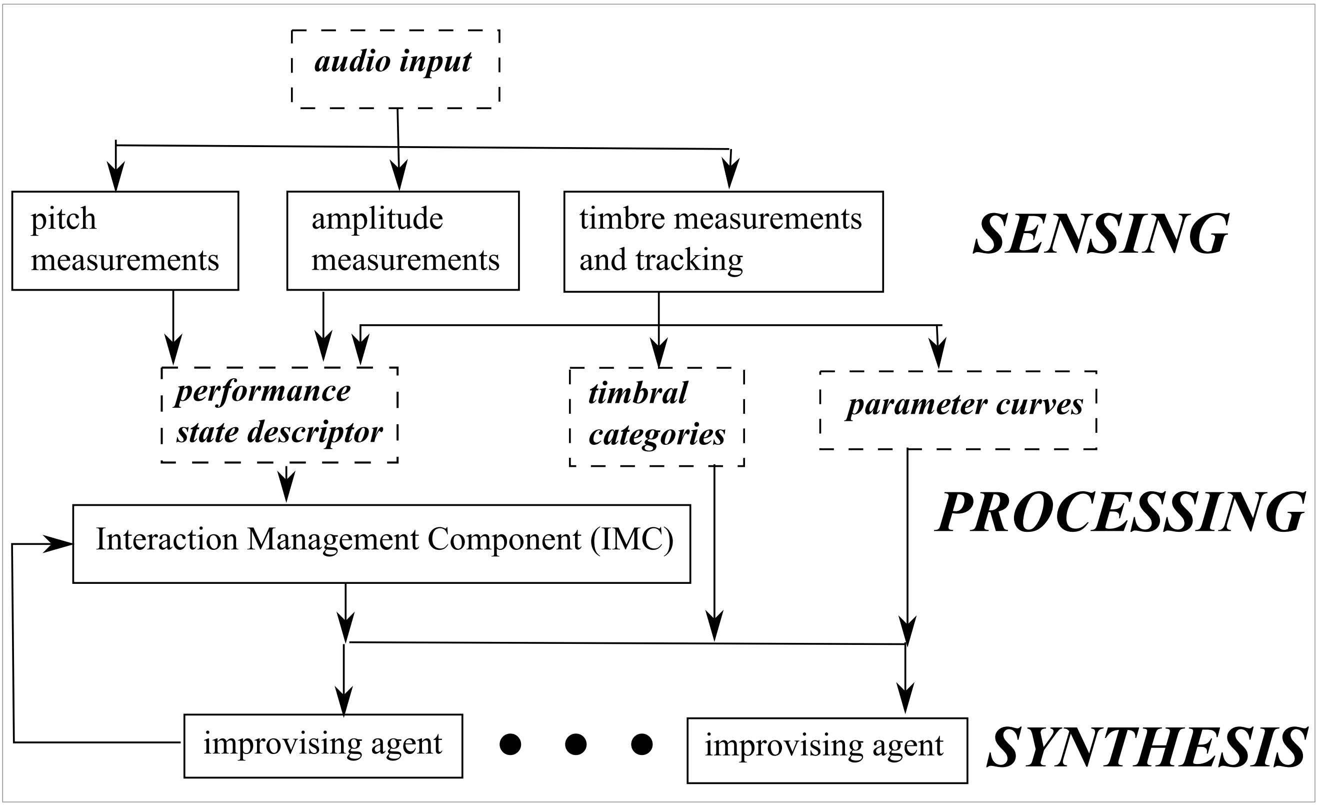


Fig. 1. Block Diagram of the ARHS System. Timbral and gestural information is extracted from the real-time audio stream (Sensing stage), analyzed by the Processing stage, and made available to a virtual improvising ensemble (Synthesis stage).

Young's NNMusic [15] works with pitch and timbral features such as brightness. It has participated in a variety of score-based and improvisatory pieces, with instruments such as flute, oboe and piano.

Casal's *Frank* [16] combines Casey's Soundspotter MPEG7 feature recognition software [17] with genetic algorithms; Frank has performed with Casal on piano and Chapman Stick.

Organizational Framework

Rowe's *Interactive Music Systems* [18] characterizes the processing chain of MIDI-oriented interactive music systems as typically consisting of an input *Sensing* stage followed by a computational *Processing* stage and an output *Response* stage. Blackwell and Young [19] discuss a similar organization.

For the purposes of this paper, I will assign operations to subsystems in the following manner. Starting at the input to a machine improvisation system, we capture real-time audio data and subdivide the data into windows. Per-window measurements and features are computed. These operations will be discussed under the Sensing stage.

At this point, the Processing stage takes data collected by the Sensing stage and organizes it into high-level descriptors of the performance state. There is usually post-processing of extracted features, determination of musical structures and patterns and the development and preparation of behavioral guidelines for improvising agents.

Finally, the outcomes of the Processing stage are fed to subsystems that directly generate audio output; this is the *Synthesis* stage. Boundaries between the stages tend to blur in real systems; however, this is a convenient framework for discussing information flow.

ARHS System Overview

The Adaptive Real-time Hierarchical Self-monitoring (ARHS) system (Fig. 1) is the latest of my machine improvisation systems. It extracts in real-time perceptually significant features of a saxophonist's timbral and gestural variations and uses this information to coordinate the performance of an ensemble of virtual improvising agents.

Timbral and gestural information is extracted from the real-time audio stream (Sensing stage) and made available to a virtual improvising ensemble (Synthesis stage). The Interaction Management Component (IMC) computes high-level descriptions of the performance (Processing stage) and coordinates the high-level behavior of the ensemble.

Sensing

The Sensing stage extracts low-level features from the input audio. These features usually include MIDI-like note events, FFT bins and various timbre measurements.

The Sensing Stage in the ARHS system

My earlier work [20] took a relatively compositional approach in emphasizing the detection and management of specific timbral and gestural events. In contrast, the ARHS system works with general event types and performance descriptors. I was interested in constructing compact, perceptually significant characterizations of the performance state from a small number of audio features. The extensive literature in mood detection and tracking from the Music Information Retrieval community mostly applied to traditional classical music or pop music (see, for example, Yang [21] and Lu et al. [22]); it suggests that compact performance state descriptors might be usable. In Lu et al. [23], three primary sets of features were found to be important for mood classification: intensity (essentially loudness/energy), rhythm (tempo, regularity, etc.) and timbral features (brightness, bandwidth, spectral flux, etc.).

For efficient real-time analysis, I chose a simplified trio of features for ARHS: loudness (loud/soft), tempo (fast/slow), and for timbre, I needed a single measure that correlates well with musical tension/release or consonance/dissonance. *Auditory roughness* has been found to constitute a significant factor in dissonance judgments [24]. Roughness quantifies interference between partials in a complex tone. My experiences with saxophone tones, with many partials and rich inter-relationships, also indicate that roughness is a useful overall measure that tracks the variation of perceptually significant characteristics. As per Vassilakis [25], roughness is estimated by extracting partials from the audio; for each pair of partials, its contribution to the roughness measure is computed, and the roughness contributions of all pairs are summed. The ARHS system utilizes FFTs and a version of Jehan's analyzer~ [26], customized to compute auditory roughness.

Figure 2 shows the main operations of the Sensing stage. The audio input is divided into windows. To form an estimate of the tempo, I segment the audio input into note events, based on pitch estimates and amplitude envelope shapes. Note onsets are determined by changes in the amplitude envelope and changes in pitch rated with a high confidence score by the pitch tracker. A roughness estimate is also computed and averaged over a sliding 1-sec window. A performance mode descriptor, comprising loudness (loud/soft), tempo (fast/slow) and timbre (rough/smooth), is derived.

The computation of roughness requires large analysis windows (>100 millisec); a fast run of notes is often classified as *rough* because of note transitions and instability within the window. Hence, audio segments classified as *fast* are not assigned a roughness descriptor. Each 1-sec segment is classified into one of seven performance modes: silence, slow/soft/smooth, slow/soft/rough, slow/loud/smooth, slow/loud/rough, fast/soft, fast/loud. From here on, the ARHS system works primarily with this compact performance state in its material selection decisions. While it is unusual for this much data reduction to be performed so early, this decision seems reasonable in light of the MIR work cited earlier.

Other Approaches

Most other machine improvisation systems take fairly similar approaches to sensing. Beginner's Mind [27] also uses measurements from Jehan's analyzer~. Van Nort's system [28] works with spectral centroid, spectral deviation, pitch strength and a textural measure (based on Linear Predictive Coding analysis). Yee-King's system [29] extracts pitch and amplitude from input audio, in addition to FFT-based spectral snapshots. NN Music [30] comprises a pitch-oriented analysis function and an audio analysis function that measures loudness, brightness, etc. Frank [31] uses vectors of log frequency cepstral coefficients generated by Soundspotter [32].

The Processing Stage

The Processing stage is where most researchers have expended the most effort, employing a variety of approaches to emulate human performance behavior. As mentioned earlier, ARHS's Sensing stage passes a compact performance descriptor to the Processing stage; a straightforward mechanism is then used for material selection and self-evaluation. In most other systems, the decision-making processes of the Processing stage work directly with a profusion of low-level information.

I have found it useful to frame a number of issues encountered here in the form of two questions. Given a high-level description of the performance state, (1) How does the system determine the timing of its performance actions? (This addresses how responsive a system is to the human's performance.) (2) How does the system select musical materials to use in its performance actions? (This addresses how the system chooses and evaluates combinations of musical materials, and whether these decisions can adapt to changing performance conditions.)

In attempts to achieve behavioral characteristics analogous to human performance (Young [33] identifies some useful attributes, such as adaptability and intimacy), several systems have incorporated neural networks and genetic algorithms in decision-making processes [34–37]. However, in the open environment of free improvisation, it is not easy to define what constitutes desirable outcomes to converge to, or fitness functions for evaluation. Yee-King and Casal [38,39] use timbral matching as a goal, but that seems rather limiting. I will expand on related issues in subsequent sections.

The Processing Stage in the ARHS System

At the beginning of the Processing stage, it is customary for machine improvisation systems to construct high-level descriptions of the human improviser's performance. A common approach is to partition the extracted feature stream into sonic gestures.

My earlier systems work extensively with feature curves organized by gesture [40]. The ARHS system is also capable of such, but takes a somewhat different approach. The ARHS Sensing stage computes a performance mode for each 1-sec window of audio. Its Processing stage uses the 1-sec descriptors and additional information to influence the timing of performance actions and make decisions on material choice.

Short-Term Responsiveness.

In an improvisation that we characterize as "responsive" or "tight," the *timing* of a particular performance action seems tied to observed events in a very small time window. (By performance actions, I mean not just simple call/response activity, but also the small timbral and gestural adjustments improvisers make when engaging in dialog.)

In the ARHS system, I define sets of *potential trigger events* (PTEs) based on short-term transition patterns in performance modes; they represent "ear-catching" events that encourage a change in the current behavior of an improvising agent. ARHS PTEs include gesture onset and end, sudden changes in performance mode, a long period of a high-intensity performance mode and the sudden occurrence of selected timbral features, for example, a sharp attack caused by a slap-tongue. An agent may respond to a PTE by starting or stopping a phrase, changing some timbral characteristic, etc. If an agent responds to a PTE, the probability that it responds to the next PTE is reduced; similarly, if it ignores the current PTE, it is more likely that it responds to the next one. For simplicity, the current ARHS implementation does not distinguish between different PTE types. I feel that this attention to relatively fine-grain timing issues, that is, decisions of *when* to play (or make an adjustment), is crucial for responsive interactive performance.

Long-Term Adaptivity.

Over an extended improvisation, I wanted to emulate interaction behavior that I enjoy in human improvisations, where a mode of dialog may be sustained, then shift significantly as each improviser takes on different roles. An agent should have some reasonable strategy for choosing a performance mode in response to a human performance mode. There should be a mechanism for evaluating a combination of human/agent performance modes and changing the mapping if necessary.

One approach that I tried to avoid is to implement a moment-to-moment mapping of the human's performance state to the system's performance state. Improvisations I enjoy are based not so much on direct mirroring (such as overly obvious echo gestures that may arise in a delay line-based system), but more on negotiation and dialog. Another problematic situation was reported by Casal [41]; a complex system may also "optimally" and consistently converge to the same output for a specific input gesture. In Young's terminology [42], there should be sufficient *Opacity* between performance state descriptors and proposed performance actions.

The ARHS system bases material selection on a fuzzy 8-sec summary of the human's performance, comprising frequencies of occurrence of each performance characteristic (fast/slow, loud/soft, rough/smooth). Each frequency is quantized roughly into frequent/moderate/seldom; hence there are 3 x 3 x 3 = 27 different performance summaries. These are mapped to performance modes for guiding agent actions. We will see later that improvising agents synthesize their performance actions, with variations based on performance mode guidelines, instead of processing the input audio stream; this further prevents any locked-in mapping of input performance gesture to output performance action.

Mechanisms are needed for evaluating combinations of musical materials in performance. The ARHS system attempts to "guess" the evaluation of the human improviser from transitions in the human's performance mode statistics. Suppose the human improviser is observed to play somewhat consistently in mode H1 over an extended period, while an agent is playing in mode A1. One might reasonably assume that the human considers such a combination desirable. If the human considers the mode combination to be a musically unacceptable clash of activity, one might reasonably expect the human to change her/his performance mode to adjust to the undesirable situation. Hence, the agent is "encouraged" to continue behavior in mode A1, if it continues to observe the human playing in mode H1. However, if the agent has been playing in mode A1, and observes the human switching to mode H2, the combination of observed mode H1 and agent mode A1 is discouraged; that of observed mode H2 and agent mode A1 is encouraged.

The Adaptive Performance Map (APM) in Fig. 3 is a table that maps each set of observed (human) performance mode statistics to a response (agent) mode. It has 27 entries, corresponding to the 27 performance summaries. Each entry contains one or more *recommended modes* for an agent. A score is kept for each mapping and is updated to indicate a preference for that mapping. Response modes with high scores are used to guide the agent's performance; modes with low scores are eventually dropped. Hence, an agent's response to a specific human performance mode may change over the course of a performance, depending on the changing preferences of the human improviser.

Summary.

It is convenient to divide the functionality of the ARHS Processing stage into two parts: *short-term timing decisions* that determine *when* to play, and *long-term material selection decisions* that determine *what* to play. Tying agent actions to Potential Trigger Events enhances temporal responsiveness. The Adaptive Performance Map enables novel combinations of musical materials in performance without an extended training session. While other machine improvisation systems may approach the problems from rather different perspectives, the division of processing into timing and material selection decisions is a useful one and will be applied for comparison with other systems.

Other Approaches

Several systems organize data from the Sensing stage into gesture-based collections. Beginner's Mind [43] segments the input audio into phrases based on loudness and attack heuristics. The mean and standard deviation of pitch and timbral measurements over a phrase are stored and used to identify the type of instrument or sound and differentiate performance styles. A fully autonomous mode was in development, but few details were given [44].

In Hsu [45], parameter curves for loudness, brightness, roughness, etc. are captured on the fly and stored at the phrase level; this retains perceptually significant input gestural shapes. These curves are used to guide the synthesis of audio output from various improvising agents, for example, a waveguide bass clarinet. Performance timing is based on detection of specific "hot button" events, such as the presence of a timbral feature.

Van Nort et al. [46] also organize feature vectors into gestures. During a training phase, representative gesture types are used to build a gesture dictionary. In performance, sonic gestures from human improvisers are matched with dictionary entries and influence the Genetic Algorithm (GA)–based evolution of a population of gestures. Different gesture types are mapped to different modes of granulation or delay-based processing. There is little discussion of more general mapping strategies or how performance timing decisions are made.

In Yee-King [47], a *Live Performance* system extracts pitch and amplitude from the input audio into a *control data store*. Pitch and amplitude sequences are used as control data for synthesizing output; it is not clear how detailed performance timing decisions are made. A *Sound Design* system attempts to match the timbre of the input audio using a GA-based approach. FFT bin analysis data from the Sensing stage represents the *target sound*; the GA attempts to generate synthesis parameters to match the target.

In Frank [48], MPEG7 feature vectors are used to influence the GA-based coevolution of populations of individuals. A fitness function, based on timbral similarity to the input sound, determines the winning individuals; the winners are presented as live sound. It is not clear how event timing is decided.

NN Music [49] accumulates statistics of loudness, brightness, etc. to compile a performance state. A library of performance states is used to train feedforward neural network A. Real-time incoming performance states are matched with the learned states in network A. A second neural network (B) generates synthesis control parameters in response to specific performance states generated by network A. Evolving stochastic processes, not documented in detail, determine the timing of sound event and rhythm generation.

In Bown and Lexer [50], Continuous-Time Recurrent Neural Networks (CTRNNs) are used to control a spectral filter, which adjusts the decay time of piano tones. Subsequent improvisations with CTRNNs involving saxophone and percussion are described briefly in Bown [51].

In the free improvisation simulation of Collins [52], he identified a large parameter set for describing the "temperament" of improvising agents that influence each other in a network. These parameters include "shyness" (amplitude of responses), "sloppiness" (degree of deviation from referenced rhythmic material) and "keenness" (probability that an agent responds to a trigger). Collins has also explored the use of *reinforcement learning* in a MIDI-based system [53]; an agent in some current performance state chooses its next action and evaluates consequent performance states. Behavioral types that indicate mutual influence are reinforced. It should be possible to incorporate timbral features into performance states and performance actions and apply similar learning mechanisms.

THE SYNTHESIS STAGE

Design decisions in the Synthesis stage tend to target-specific musical styles and situations; hence, they are relatively ad hoc compared with the Sensing and Processing stages. Collins for example has commented on attention to timbral choice [54], which can be important for achieving good musical results.

The Synthesis Stage in the ARHS System

In the ARHS system, the Synthesis stage comprises one or more improvising agents. Each agent has access to output from an Adaptive Performance Map (APM), with high-level guidelines on material selection (fast/slow, loud/soft, rough/smooth). As the human improviser begins to play, potential trigger events (PTEs) are generated. Eventually, an agent activates, consults its APM for a performance mode and synthesizes a gesture based on mode guidelines. As the performance develops, additional PTEs encourage adjustments to the agent's ongoing gesture, and the APM's score for the current combination of human/agent performance mode is updated. Human/agent performance modes with low scores may be discarded and mappings altered. Eventually, the human's performance may change significantly. When the agent begins a new gesture, it will consult its APM to select a response performance mode with a good score and synthesize the new gesture.

Hence, agents can learn workable material combinations during performance and adjust them through an extended improvisation. It may seem restrictive to set an implicit goal of maintaining stable combinations of observed and response modes. An agent can also decide to change its response mode out of "boredom" with an overly extended period of the same mode combination, or choose to specifically defeat the goal of maintaining consistent mode combinations; these are options for future exploration.

As in my earlier systems, each ARHS agent "plays" a virtual instrument by generating parameter curves that control loudness and timbral features such as brightness, roughness (usually a tremolo envelope), etc. and using these curves to synthesize output sound. Pitch tends to be less important in free improvisation; consequently, pitch is handled in a straightforward way, by extracting the (occasional) stable pitch from the input audio and applying simple transformations. This general approach works well with a variety of synthesis algorithms, from simple filtered noise sources, to waveguide-based clarinets, to metallic-sounding comb filters.

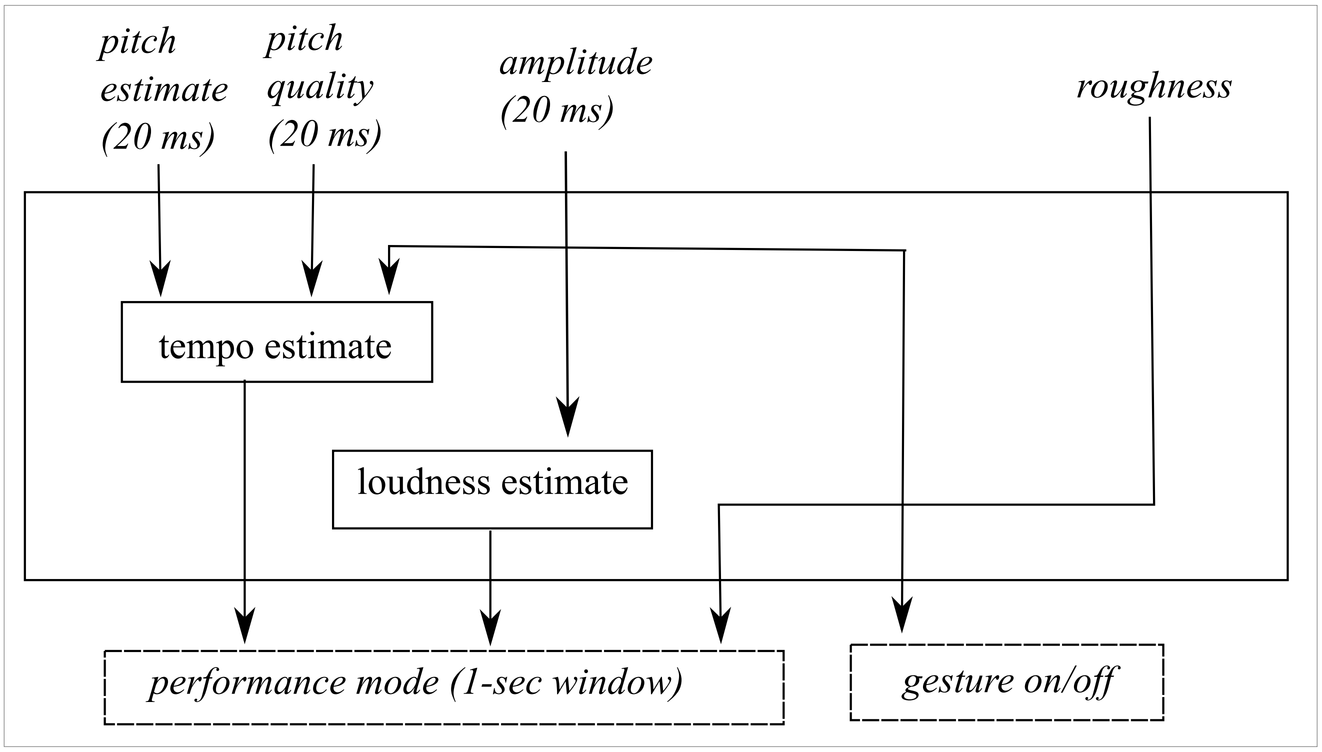
Other Approaches

Many of the systems overviewed here are relatively vague on specifics of the Synthesis stage. Most involve adaptive processing of the input or synthesis of response materials.

Of the processing-based systems, *Beginner's Mind* [55] runs the input audio through a reconfigurable signal-processing network including ring modulation, FFT-based analysis/resynthesis, delay lines and granulators. Van Nort's system [56] uses collections of delay lines and granulators to process the input audio; processing parameters are influenced by gestural length and gestural type.

Of the synthesis-based systems, NNMu-

Fig. 2. Sensing Stage of ARHS system. In the Sensing stage, the audio input is divided into windows.



sic [57] generates MIDI data to control a Disklavier or control data for the playback and transformation of samples. In Yee-King's system [58], pitch/amplitude sequences generated in the Processing stage control an additive or FM synthesizer. Frank [59] controls playback of sequences of MPEG7 frames, which can be sourced from previously recorded sound, or from live performance. Collins's free improvisation simulation for guitar and agents [60] works primarily with onsets, pitch and other note-level events. Agents play Karplus-Strong and comb filter synthesis modules to blend with the human musician's guitar sound.

EVALUATION

The ARHS system is an attempt to encode and emulate some aspects of the musical knowledge and performance practices of my collaborators and myself. It has performed in concerts with John Butcher and James Fei. As with other machine-improvisation systems overviewed in this paper, the sonic outcomes are the results of layers of adaptive algorithms and random processes, all in real-time interplay with the human musician. The success of such experiments is difficult to assess objectively.

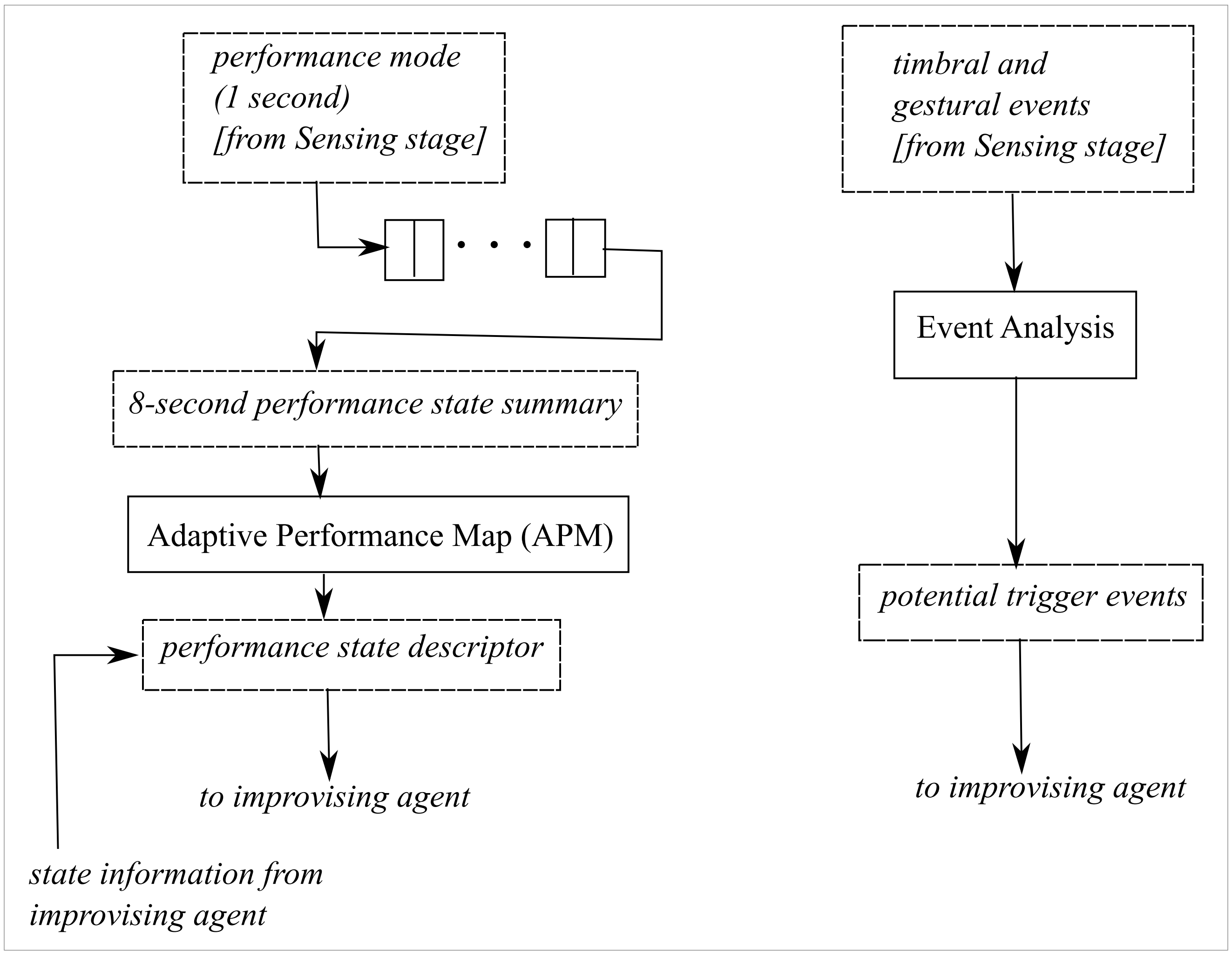
Collins, among others, has observed that the evaluation of interactive music systems (IMSs) "has often been inadequately covered in existing reports. . . ." [61] He proposed three suggestions for evaluating IMSs: (1) Technical criteria related to tracking success or cognitive modeling; (2) The reaction of an audience; (3) The sense of interaction for the musicians who participate.

It is perhaps not so useful to compare disparate systems such as Voyager and Frank, grounded as they are in specific musical practices and situations. Rigorous evaluation procedures are useful, however, for comparing similar systems, to aid in the selection of specific components or subsystem implementations, or fine-tuning of configurations and parameters, as part of the design process.

A detailed discussion of the salient issues is beyond the scope of this paper. Hsu and Sosnick [62] is an attempt to identify major issues and develop rigorous procedures for testing and evaluation from both the performer's and listener's points of view; the concepts and procedures developed were applied to compare the ARHS system and its predecessor, the London system.

Ultimately, we are interested in whether a proposed combination of strategies and implementations leads to interesting results, in a particular musical context. Rigorous human computer interaction–based evaluation methodologies are tricky to apply to the dynamic user/audience environment of a machine improvisation. In lieu of vague reflections and generalizations, the reader is invited to sample the extensive audio documentation at <userwww.sfsu.edu/~whsu/improvsurvey.html>.

Fig. 3. Processing stage of ARHS system. The Adaptive Performance Map is a data structure that maps each set of observed (human) performance mode statistics to a response (agent) mode.


performance mode (1 second) [from Sensing stage]
timbral and gestural events [from Sensing stage]
Event Analysis
8-second performance state summary
Adaptive Performance Map (APM)
potential trigger events
performance state descriptor
to improvising agent
to improvising agent
state information from improvising agent

References and Notes

1. R. Rowe, *Interactive Music Systems* (Cambridge, MA: MIT Press 1993).

2. T. Blackwell, and M. Young, "Self-Organised Music," *Organised Sound* **9**(2), 123–136 (2004).

3. G. Lewis, "Too Many Notes: Computers, Complexity and Culture in *Voyager*," *Leonardo Music Journal* **10** (2000).

4. T. Jehan and B. Schoner, "An Audio-Driven Perceptually Meaningful Timbre Synthesizer," *Proceedings of the International Computer Music Conference* (2001).

5. M. Malt and E. Jourdan, "Real-Time Uses of Low Level Sound Descriptors as Event Detection Functions Using the Max/MSP Zsa. Descriptors Library," in *Proceedings of the 12th Brazilian Symposium on Computer Music* (September 2009).

6. W. Hsu, "Managing Gesture and Timbre for Analysis and Instrument Control in an Interactive Environment," *Proceedings of the International Conference on New Interfaces for Musical Expression,* Paris, France (2006).

7. W. Hsu, "Two Approaches for Interaction Management in Timbre-Aware Improvisation Systems," *Proceedings of the International Computer Music Conference,* Belfast, U.K. (2008).

8. P. Clark, "Breaking the Sound Barrier," *The Wire* (March 2008). See also <www.johnbutcher.org.uk>.

9. N. Collins and J. d'Escrivan, eds., *The Cambridge Companion to Electronic Music* (Cambridge, U.K.: Cambridge Univ. Press, 2007) p. 182.

10. T. Ciufo, "Design Concepts and Control Strategies for Interactive Improvisational Music Systems," *Proceedings of MAXIS International Festival/Symposium of Sound and Experimental Music,* Leeds, U.K. (2003).

11. T. Ciufo, "Beginner's Mind: An Environment for Sonic Improvisation," *Proceedings of the International Computer Music Conference* (2005).

12. N. Collins, "Towards Autonomous Agents for Live Computer Music: Realtime Machine Listening and Interactive Music Systems," Ph.D. Thesis, Univ. of Cambridge, Cambridge, U.K., 2006.

13. M. Yee-King, "An Automated Music Improviser Using a Genetic Algorithm Driven Synthesis Engine," EvoMusart workshop, evo* 2007, published in *Applications of Evolutionary Computing,* **4448** of *Lecture Notes in Computer Science* (LNCS 4448) (Springer, April 2007) pp. 567–577.

14. D. Van Nort et al., "A System for Musical Improvisation Combining Sonic Gesture Recognition and Genetic Algorithms," *Proceedings of Sound and Music Computing Conference,* Porto, Portugal (2009).

15. M. Young, "NN Music: Improvising with a 'Living' Computer," in R. Kronland-Martinet et al., eds., *Computer Music Modelling and Retrieval: Sense of Sounds,* Lecture Notes in Computer Science Series (Springer-Verlag, 2008).

16. D. Casal, "Time after Time: Short-circuiting the Emotional Distance Between Algorithm and Human Improvisation," *Proceedings of the International Computer Music Conference,* Belfast, U.K. (2008).

17. M. Casey, "Soundspotting: A New Kind of Process?" in R. Dean, ed., *The Oxford Handbook of Computer Music* (New York: Oxford Univ. Press, 2009).

18. Rowe [1].

19. Blackwell and Young [2].

20. Hsu [6].

21. Y. Yang, "Music Emotion Classification: A Fuzzy Approach," in *Proceedings of ACM Multimedia,* Santa Barbara, CA (2006) pp. 81–84.

22. L. Lu et al., "Automatic Mood Detection and Tracking of Music Audio Signals," in *IEEE Transactions on Audio, Speech, and Language Processing* **14**, No. 1 (2006) pp. 5–18.

23. Lu et al [22].

24. Vassilakis, P., "Auditory roughness as a Means of Musical Expression," in *Selected Reports in Ethnomusicology,* 12: 119-144.

25. Vassilakis [24].

26. Jehan and Schoner [4].

27. Ciufo [11].

28. Van Nort et al. [14].

29. Yee-King [13].

30. Young [15].

31. Casal [16].

32. Casey [17].

33. Young [15].

34. Yee-King [13].

35. Van Nort et al. [14].

36. Young [15].

37. Casal [16].

38. Yee-King [13].

39. Casal [16].

40. Hsu [6].

41. Casal [16].

42. Young [15].

43. Ciufo [11].

44. Ciufo [11].

45. Hsu [6].

46. Van Nort [14].

47. Yee-King [13].

48. Casal [16].

49. Young [15].

50. O. Bown and S. Lexer, "Continuous-Time Recurrent Neural Networks for Generative and Interactive Musical Performance," EvoMusArt Workshop, EuroGP, Budapest (2006).

51. O. Bown, <www.olliebown.com/main_blog/?p=271>.

52. Collins [12].

53. N. Collins, "Reinforcement Learning for Live Musical Agents," in *Proceedings of International Computer Music Conference* (2008).

54. Collins [12] p. 180.

55. Ciufo [11].

56. Van Nort et al. [14].

57. Young [15].

58. Yee-King [13].

59. Casal [16].

60. Collins [12].

61. Collins [9] p. 183.

62. W. Hsu and M. Sosnick, "Evaluating Interactive Music Systems: An HCI Approach," *Proceedings of the International Conference on New Interfaces for Musical Expression,* Pittsburgh, U.S.A. (2009).

Manuscript Received 1 January 2010.

William Hsu works with electronics and real-time animation in performance. He is interested in building real-time systems that evoke some of the live, tactile qualities of musicians playing acoustic instruments and the evolving, unstable forms and movement observed in natural processes. He has performed in the United States, Asia and Europe; recent collaborators include Peter van Bergen, John Butcher, James Fei, Matt Heckert and Lynn Hershman. Since 1992, he has been with the Department of Computer Science at San Francisco State University, where he teaches and does research in computer music, performance evaluation and high-performance computing.

CALL FOR PAPERS

ArtScience: The Essential Connection

Guest Editor: Robert Root-Bernstein

What is the value of artistic practices, techniques, inventions, aesthetics and knowledge for the working scientist? What is the value of scientific practices, techniques, inventions, aesthetics and knowledge for the artist? When does art become science and science, art? Or are these categories useless at their boundaries and intersections?

Can an individual excel at both science and art, or is even a passing familiarity with one sufficient to influence the other significantly? Do the arts ever contribute significantly to scientific progress? Where will current scientific innovations lead the arts in the next few decades?

Submissions exploring these questions can be from artistic scientists who find their art avocation valuable; from scientist-artist collaborators who can demonstrate a scientific or artistic innovation; from scientifically literate artists who draw problems, materials, techniques or processes from the sciences; or from historians of art or science looking at past examples of such interactions.

Interested authors are invited to send proposals, queries and/or manuscripts to the Leonardo editorial office: Leonardo, 211 Sutter St., Suite 501, San Francisco, CA 94108, U.S.A. E-mail: <isast@leonardo.info>.