INTRODUCING VIDEO FEATURES AND SPECTRAL DESCRIPTORS IN THE OMAX IMPROVISATION SYSTEM

Georges Bloch

Ircam – CNRS UMR 9912,
Paris, France,
Université Marc-Bloch,
Strasbourg, France
Georges.Bloch@ircam.fr

Shlomo Dubnov

UCSD
9500 Gilman Dr. MC 0326
La Jolla, CA 92093-0326
sdubnov@ucsd.edu

Gérard Assayag

Ircam – CNRS UMR 9912,
Paris, France,
assayag@ircam.fr

ABSTRACT

Some decisive features have been added to the OMax computer assisted improvisation system: one is a video interface, which makes the virtual clone of the performer not only audible but visible. Another one is a drastic extension of the symbolic world on which the improvisation is built: to an event-driven (basically pianoroll-like) representation has been added another one using spectral descriptors. The results are particularly exciting for a number of musical cases (timbre oriented music of course, but not only). It adds to the generality of an already agnostic system. The relationship of OMax to real-time, especially in follow mode, has to be noted in case of a music performed on a network. As for the video features, they allow a kind of sound driven total performance, in which written parts (with eventual filmed pictures) can be the pretext to free improvisatory moments intimately related to them. The fact OMax can continue an existing session (improvised or written) opens the door to all kind of hybrid interconnections between what is written, what is improvised, what is already existing and what is to be.

1. INTRODUCTION

The OMax improvisation system has been developed at Ircam in the recent years [3, 13]. It has been used in performances with such famous improvisers as Bernard Lubat or Mike Garson and is under constant improvement with the expertise of great musicians.

The OMax project followed an earlier project on style modelling introduced at Ircam by Shlomo Dubnov [9]. Marc Chemillier, Gerard Assayag and Georges Bloch worked on a real-time interactive version, where the modeling process happens quickly enough so as to allow immediate generation as a response to the live musician. The artefact produced by the machine can be seen as a « musical clone » of the musician. The first version was pure Midi, then came an audio version based on an extension of the Yin [7] pitch follower. Finally, this paper describes the recent addition of the video clone module and the extension of sound feature extraction to generalized spectral descriptors.

2. OMAX ARCHITECTURE OVERVIEW

The OMax architecture stands over two well-known environments : Max and OpenMusic. The Max patch handles real-time interaction: it captures sound, extracts features, segments, and packs this information into a stream of OSC messages sent to OM. These messages contain a data structure called « Augmented Midi Unit» (AMU). OM learns this data into a stylistic model. Playing the model results in sending back to the Max patch a stream of AMU through OSC. Max processes this data in order to render the machine improvisation. It can do so in Midi, or using the recorded instrumental sound through a concatenative-synthesis like process, or it can use the symbolic parameters received from OM to control a synthesis engine or a virtual acoustics processor. The OM part hosts a collection of concurrent agents responsible, among other tasks, for the learning and improvising processes. Learning involves an algorithm called the Factor Oracle [1, 5, 8] that builds a sequential graph over the sequence of AMU received from Max (see Figure 1).

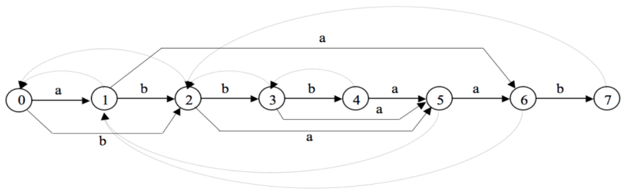

Figure 1. The Factor Oracle for string abbbaab.

In this graph, the nodes are states, each edge is labelled with an AMU that describes a consecutive segment of the instrumental source : it could be a note, a chord, a pointer to a location in the live recording, descriptors for a spectral frame or a series of contiguous frames, or a combination of those. The backward arrows called suffix links connect maximally repeating patterns with respect to a chosen classification function. By a sophisticated heuristics of navigation through this structure [2] a stylistically similar improvisation is generated, packed in a stream of AMU and sent back to Max who renders it.

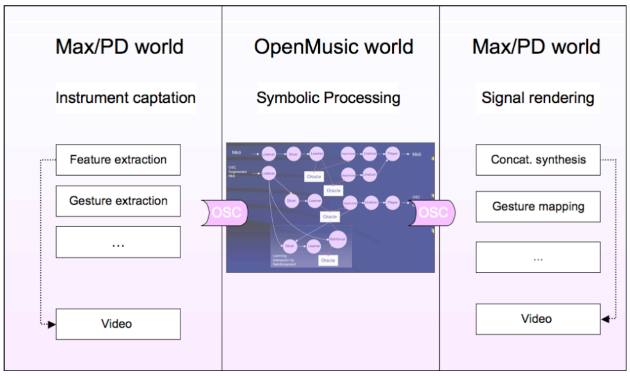

Figure 2. OMax architecture.

The respective roles of OpenMusic and Max are summarized in Figure 2. As Figure 2 shows, the video data (as well as the actual sound datas captured in the performance) are subservient to signal extraction feature, which deduces some kind of symblic representation.

3. THREE LEVELS OF REPRESENTATION

OMax plays on three levels of representation. It actually operates on the AMU level – the symbolic level of musical representation – when the result of these operations are applied to two other levels: sound and video (in most cases they can also be directly transformed into MIDI data). The current version offers a choice between two kinds of musical descriptors: notes (pianoroll or MIDI-like notation) and spectral descriptors. In some way the term "Augmented Midi" could be misleading, since spectral descriptors have very little to do with MIDI. However, for the sake of convenience, we'll continue using the term AMU to refer to the symbolic level unit.

3.1. The Symbolic level is the boss

The most important feature is the priority given to this symbolic level. The factor oracle reorganize the input according to the information retrieved at this level, and all other levels of representation depend on it. More precisely, they are simply indexed to the symbolic extraction. Whatever the symbolic representation used, an index to the audio and video streams is added to it. In case of a pianoroll representation, it really looks like a MIDI vector with an absolute date and reference date to the sound and video buffers added to the pitch, velocity and duration.

Concerning video, this means a non trivial consequence: the improvisation is built on the sonic features and conducts the pictures. It is a sound-driven software with video capacities. This actually is quite an original feature in our image-driven world.

3.2. Sound and video architecture

The sound of the improviser is fed into the system: although it can be realised with one single input, it works better with two inputs, one for symbolic detection and one for recording. Low dynamic contact mikes are better when used for the symbolic detection: not only it does impead feedback, but the low-dynamic / small frequency range actually simplifies the detection process (both in event or spectral oriented types of representation), allowing simpler equivalence classes for the oracle. For the recording itself, on the contrary, the best quality is of course mandatory.

3.3. Buffers

In the actual implementation, the sound and video are stored into buffers, and two laptop computers are used. However, some successful attemps have been made using only one laptop, and using films on files. In one case, previously filmed images (on file) and images taken "on the fly" have been used simultaneously.

The video feature buffer is timewise parallel to the sound buffer: therefore, the same date index is used for both media.

3.4. Timeline, blocks and simultaneous features

Both buffers are therefore represented with the same timeline, appearing in the user interface (see Figure 3)

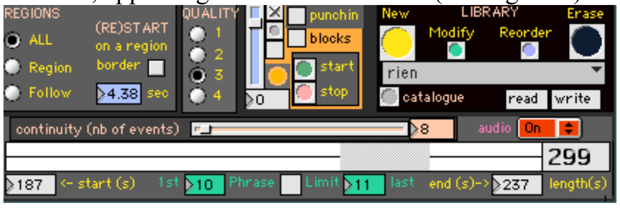

Figure 3. OMax timeline; in the upper left, the improvisation mode in relation to time can be chosen.

The number on the right gives the current length in seconds of the session, that is the live recorded material on which the system is building its improvisation. As the musician goes on playing, this duration increases, and the line below corresponds to the entire timeline from time 0 to "now". It is possible to chose a region in this timeline, either in seconds (in the Figure above, from 187 to 237) or in phrase numbers, "phrases" simply being separated by moments where the performer stops playing for some time (assignable by the user).

The chosen region can be taken either as the start of an improvisation that otherwise may navigate in the whole timeline, (mode ALL, as in Figure 3), or the improvisation can be restricted to the region (mode Region). An interesting feature in real-time is the Follow mode: the improvisation material is restricted to a moving time window of a chosen size following the performer by a given interval of time. This can be very reactive (0.5 sec), or just improvising on the last six or seven seconds. It also can be very long. The start-stop button in Figure 3 commands a "continuous" or main improvisation unit, that can go on for ever.

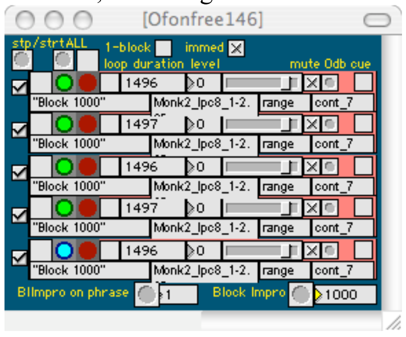

Figure 4. Blocks, for keeping patterns, or looping.

Some improvisation patterns can also be generated and kept under the sleeve for further use. This is the purpose of the blocks (see Figure 4). Five improvisations of predetermined length can be computed, kept and played independantly with the main one. They also can be looped.

3.5. Video interface

The video interface has been implemented using the Jitter environment. It allows a mixing of the main voice and the blocks, with a kind of video mixing console (see Figure 5). The bottom orange part is the capture / recording part. The upper (green part) is the video console. In Figure 5, the continuous (main) improvisation (top right window) is chosen: consequently, it is this window which appears on the main (larger) screen.

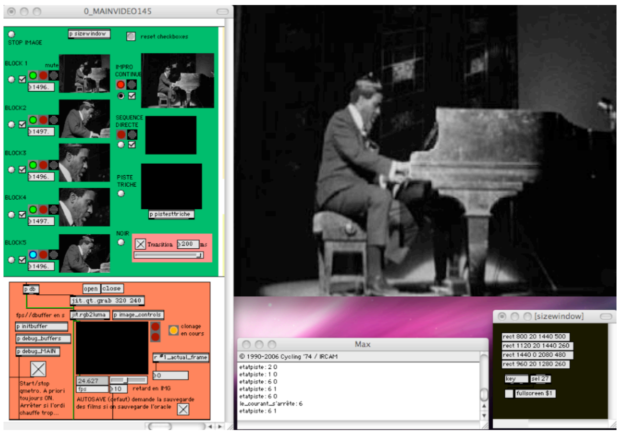

Figure 5. Video interface. On the left side: capture (bottom, orange), mixing (top green). Right: the main video window, generally projected on a large screen.

The capture is made in real-time and the improvisation can start immediately. As for the oracles, the films can be saved and recalled. Moreover, one can recall images unrelated to the sound sequence and have them "dance" on the generated improvisation.The films are now in black and white. The use of color films could rapidly be available with more powerful machines.

4. THE SYMBOLIC LEVELS: PIANOROLL AND SPECTRAL DESCRIPTORS

Historically, OMax has been first implemented in MIDI, then in audio and finally in video.

4.1. Fine pitch-tracking for a pianoroll representation

The passage to an audio OMax demanded a fine pitch-tracking system: particularly important was the timing of the events, even more that the actual pitch detected. The detected pitch could be wrong, as long as it is consistent, whereas the timing of the event has to be right, since this is what determines the date of the sound (and possibly video) buffer in the resynthesis stage.

An notable feature of yin_GB algorithm (described in [4] and based on the Yin pitch detector [7]) is its time precision and its flexibility: its presets include sound poetry and percussion. However, this pitch detection process induces a supplementary known delay of about 60 ms. This is not very important given the principle of reactivity of the virtual improvizer (around half a second, not much more than a real improviser). Some acoustic instrument can have specific pitch detector added to them, like electric guitar or piano. In this case, the MIDI input is synchronized to the actual sound recording. In any case, the resulting symbolic representation is a sort of pianoroll of polyphonic slices [3], each slice being considered as an event.

4.2. Spectral descriptors

In continuation of recent works of Shlomo Dubnov and Arshia Cont [8], a spectral version of the symbolic level has been implemented.

The purpose of having factor oracles operate on spectral features is that it allows capturing repeated sequential patterns in musical signals that do not necessarily have pitch as their dominant descriptive feature or musical parameter. Examples of such audio signals are polyphonic music, electronic music, timbral / phonetic sequences in vocal signals, as well as musical gestures in extended instrumental or vocal techniques and randomly varying natural sound textures.

Currently we are using two types of timbral features, namely the MFCC (Mel Frequency cepstrum Coefficient) and AR (Auto-regressive) coefficients. Both features are commonly used in speech recognition [12]. MFCC have been used by music researchers for music retrieval and summarization and had been shown to be statistically optimal low dimensional representations of musical signals [10]. AR features are an alternative set of features that represents a smooth version of the sound spectrum that captures the spectral envelope and discards detailed spectral information due to pitch or spectral lines. These feature capture the dominant resonance regions (formant frequencies in case of speech), and thus efficiently represent frequency areas where most of the signal energy is concentrated.

It should be noted that our approach of looking for sequential similarity in musical features is more general then the two specific features presented above. Additional features, such as chroma, beat spectrum or rhythmic features will allow "listening" to different aspects of musical processes. One interesting property of the spectral features is that in many cases repetitions of musical materials are correlated to spectral repetitions. Due to the sequential matching nature of the algorithm, recombination of sequences with similar spectrum in many cases corresponds also to meaningful repetition in pitch contents, even though tone information is not included in the representation.

4.2.1. Equivalence classes

As the factor oracle functions with distinct symbols, the results of the spectral analysis needs to be transformed in a relatively small collection of values corresponding to spectral equivalence classes. This is done by the way of a relatively severe quantisation. The quantisation factor actually is a parameter of the analysis.

All spectral analyses is implemented through the Gabor/FTM environment [11]. The auto-regressive features use lpc algorithm (see Figure 6).

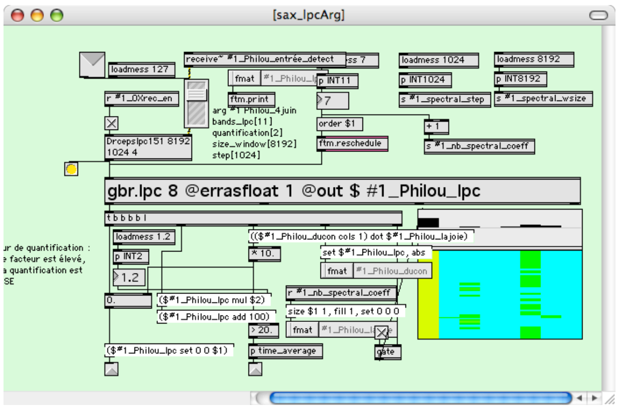

Figure 6. Gabor/FTM lpc spectral descriptor and its quantification unit.

4.2.2. Frame or event?

One non-trivial heuristic feature of the implementation is that, by quantizing the results, one often finds in the end several contiguous similar frames.These similar frames are considered as one event of longer duration, so that “frames” are not really frames, but rather events changing along the spectral evolution. Therefore the temporal heuristics (involving duration relationship) used in the event-driven version of OMax [2] still apply.

5. CONCLUSION IN FORM OF AESTHETIC REMARKS

5.1. A Large Range of musical styles

The coexistence of two kinds of representation makes OMax a most versatile system; the original extended MIDI representation, with its slicing system, is already extremely agnostic, since it does not refer to any specific grammatical feature related to any given musical style; however, it is event oriented. The spectral descriptors are more flux oriented, and work better in continuous, timbral-oriented kind of music.

5.2. The Continuation of already existing repertoire

Archived OMax sessions can be fed into the system before starting to play. An already existing repertoire can be “oraclised”. Not only can a new improvisation be extracted from it, but a musician can take it as a point of departure, as shown in Figure 5.

5.3. Composed improvisation

Current research deals on the way of finding and imposing long term structures (notably using constraint-based programs). The strength of the oracle as an improvisation system is that it considers the whole domain of the already played music and grows as the played music grows. However, large term structures can be injected into the OMax system, for example, when written movements of a piece get “oraclised” and becomes the basis for an improvisatory movement [6].

5.4. Real-time? The example of a network jam session

The relationship of OMax with real-time is very interesting. Real-time is a context variable term. It can mean “faster than sound”, faster than gesture or just reasonably fast. It seems often forgotten that a very competent real musicain does not recognize a pitch, for example, in real time, would it be just for the time for this pitch to emerge from the transient features of the instrument and the distance between the listening musician and the instrument. OMax is not hard real-time, would it be just for its Lisp-written oracle engine. However, it is quite reactive and, in follow mode, gives a reasonable clone to what is just being played. It is a good candidate for network jam session: instead of compromisong with the sound to get immediate response, better have a local clone give a good idea of what has just been played afar.

5.5. A total performance setup

The video features allow for a complete performance setup. In *Peking Duck Soup*, a piece written by Georges Bloch and premiered in Strasbourg in February 2008, a pianist played with her (filmed) miror image, like Groucho and Harpo in the almost homonyme film. The multiplied image of the player and her mirror in the subsequent section becomes troubling, when the improvising clone is itself mirrored by its filmed image.

6. REFERENCES

[1] Allauzen C., Crochemore M., Raffinot M., “Factor oracle: a new structure for pattern matching”, *Proceedings of SOFSEM'99*, Lecture Notes in Computer Science pp. 291--306, Springer-Verlag, Berlin, 1999.

[2] Assayag, G., Bloch, G., “Navigating the Oracle : a Heuristic Approach.” *Proceedings of the ICMC ’07*, pp. 405-412, Copenhagen, 2007.

[3] Assayag, G., Bloch, G., Chemillier, M., Cont, A., Dubnov, S., “Omax Brothers: a Dynamic Topology of Agents for Improvization Learning”, *ACM Multimedia 2006*, Santa Barbara, 2006

[4] G. Assayag, G. Bloch, M. Chemillier « OMax-Ofon », *Proceedings of Sound and Music Computing 2006*, Marseille, 2006.

[5] Assayag, G., Dubnov, S., “Using Factor Oracles for Machine Improvization”, *Soft Computing 8*, pp. 1432-7643, September 2004.

[6] Bloch, G., Chabot X., Dannenberg, R., “A Workstation in Live Performance: Composed Improvization”, *Proceedings of ICMC’86*, The Hague, Netherlands, 1986.

[7] de Cheveigné, A., Kawahara, H. "YIN, a fundamental frequency estimator for speech and music", *J. Acoust. Soc. Am. 111, 1917-1930,* 2002.

[8] S. Dubnov, G. Assayag, A. Cont, “Audio Oracle: A New Algorithm for Fast Learning of Audio Structures”, *Proceedings of ICMC’07*, Copenhagen 2007.

[9] S. Dubnov, S., Assayag, G., Lartillot, O., Bejerano, G., “Using Machine-Learning Methods for Musical Style Modeling”, *IEEE Computer*, Vol. 10, n° 38, p.73-80, October 2003.

[10] Logan, B. “Mel frequency cepstral coefficients for music modeling.” *Proceeding of ISMIR 2000*, Plymouth MA, 2000

[11] N. Schnell, D. Schwarz « Gabor, Multi-Representation Real-Time Analysis/Synthesis », *COST-G6, DAFx’05*, Madrid, 2005.

[12] Rabiner L., Juang B.H., *Fundamentals of Speech Recognition*, Prentice Hall, 1993.

[13] http://www.ircam.fr/equipes/repmus/OMax