

Concatenative Synthesis of Expressive Saxophone Performance

Stefan Kersten*, Rafael Ramirez*

*Music Technology Group, Universitat Pompeu-Fabra, Barcelona, Spain

Abstract—In this paper we present a systematic approach to applying expressive performance models to non-expressive score transcriptions and synthesizing the results by means of concatenative synthesis. Expressive performance models are built from score transcriptions and recorded performances by means of decision tree rule induction, and those models are used both to transform inexpressive input scores and to guide the concatenative synthesizer unit selection.

I. INTRODUCTION

In the past, important approaches to expressive performance modeling have been empirical methods based on statistical analysis, mathematical modeling, and “analysis-by-synthesis” (see the summary provided in [1]). In all these approaches, it is a person who is responsible for devising a theory or a mathematical model which captures different aspects of musical expressive performance. The theory or model is later tested on real performance data in order to determine its accuracy.

Our approach as well as the one described in [1] is based on building computational models of expressive performance by machine learning, in our case inductive logic decision tree models. Those models are used to predict expressive transformations for inexpressive scores as well as guiding sample database note selection and transition modeling in a concatenative saxophone synthesizer.

II. EXPRESSIVE PERFORMANCE ANALYSIS

Figure 1 gives an overview of the functional parts of the expressive performance analysis and synthesis system.

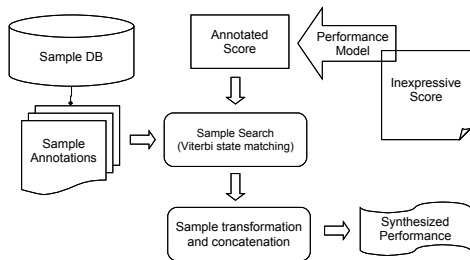


Fig. 1. Expressive performance analysis and synthesis system overview.

Modeling of expressive performance involves analysis of recorded performances of a musical piece and the comparison of the symbolical information extracted with the one present in the score in order to build a computational

model for a particular performer in a particular style of music.

Our approach can be divided into three distinct steps: In a preprocessing stage low- and high-level perceptive features are extracted from a short-time fourier transform (STFT) representation of the musical audio recording and are grouped into a note-level transcription (Fig. 2). The transcribed representation is aligned to the symbolic score in a second step and finally a computational model of the performance’s characteristics is built.

A. Note segmentation and feature extraction

In this section we shall be concerned with extracting the symbolic note-level descriptors from a performance recording that are needed both for building the computational performance model and for synthesizing a score enriched with expressivity annotations.

Most of the features are calculated from a short-time fourier transition (STFT) signal representation, i.e. overlapping frames of time-domain audio data that are multiplied by a window function and transformed to the frequency domain by the discrete fourier transform (DFT). In our descriptor database we used a frame size of 1024 with an overlap of 50% at a sample rate of 44100 kHz. The window function used is a Kaiser-Bessel window [2] with a 25dB side-lobe to main-lobe ratio.

The main low-level features used for describing expressive performance are fundamental frequency and mean energy. Log-mean-energy is extracted from a time domain representation using frames of size 1024 with a 50% overlap weighted by a Blackman-Harris window w of length N according to (1).

$$E_n = 20 \log \frac{\sum_{m=-\infty}^{m=\infty} [x(m)w(n-m)]^2}{N} \quad (1)$$

Instantaneous fundamental frequency is extracted following the two-way mismatch procedure described in [3] and [4]. In order to obtain a “brightness” measure, the spectral centroid of the discrete spectrum $X(n)$ of size N is extracted according to (2).

$$F_{centroid} = \frac{\sum_{n=0}^{n=N-1} f(n)|X(n)|}{\sum_{n=0}^{n=N-1} |X(n)|} \quad (2)$$

Note segmentation is performed in a two-step algorithm based on descriptors extracted from the spectral signal representation. First, an onset function is calculated based on energy envelopes in different bands and by applying

Note onset
 Note offset
 Note duration
 Fundamental Frequency
 Mean energy
 Energy envelope attack time
 Energy envelope sustain init level
 Energy envelope sustain end level
 Energy envelope sustain time
 Energy envelope release time
 Legato left
 Legato right
 Mean spectral centroid

TABLE I
 FEATURES EXTRACTED DURING PRE-PROCESSING

psycho-acoustical knowledge [5]. In a second step, the onset function is combined with a pitch-transition function to yield the final note onset and offset detection function.

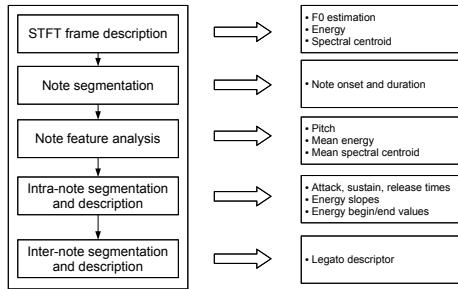


Fig. 2. Feature extraction for Expressive Performance Analysis.

The note-level feature j is calculated from the frame-based feature values by calculating the mean value over the analysis frame indices contained in the i th note's onset and offset times expressed in terms of analysis frames k_{on} and k_{off} (3). In the case of note fundamental frequency and pitch estimation a histogramming approach according to [6] is employed, in order to smooth out large errors in instantaneous fundamental frequency extraction.

$$\bar{F}_j(i) = \frac{\sum_{k=k_{\text{on}}}^{k_{\text{off}}} F_j(k)}{k_{\text{off}} - k_{\text{on}}} \quad (3)$$

Once note onsets and offsets have been determined, energy envelope attack and release times and the corresponding energy levels are extracted along with a *legato* descriptor, that captures the “smoothness” of transition between two successive notes [7]. Table I lists all of the features being used either by the model generation step, the concatenative synthesizer or both.

Notes –or *units*– extracted from performance audio files are organized in a file-system based database containing phrase- and note-level information in XML files and accompanying PCM audio files and binary analysis files used by the synthesizer.

B. Modeling of expressive performance

The symbolic representation obtained in the analysis step is aligned to the symbolic score by dynamic

timewarping and subsequent manual resolution of errors. Since a score note only contains a limited amount of useful information for expressivity analysis, an attempt was made to capture and associate more meaningful musical context with each individual score note. For this purpose the Implication/Realization model described in [8] is used, which describes relationships within a set of notes with regard to registral direction and intervallic difference.

The principle of registral direction states that small intervals imply a following interval in the same registral direction (a small upward interval implies another upward interval and analogously for downward intervals), and large intervals imply a change in registral direction (a large upward interval implies a downward interval and analogously for downward intervals). Based on these two principles, melodic patterns or groups can be identified that either satisfy or violate the implication as predicted by the principles.



Fig. 3. Prototypical Narmour structures



Fig. 4. Prototypical Narmour structures

Figure 3 shows prototypical Narmour structures. A note in a melody often belongs to more than one structure, i.e. a description of a melody as a sequence of Narmour structures consists of a list of overlapping structures. We parse each melody in the training data in order to automatically generate an implication/realization analysis. Figure 4 shows the analysis for a fragment of the Jazz standard *All of me*.

The resulting aligned sequences are cast into several inductive logic models (see II-C), one for note onset prediction, one for note duration prediction and one for note transition (legato) prediction. The models –capturing predictions for note onset and duration transformations– can be applied to a non-expressive input score, e.g. a MIDI or MusicXML description, to obtain an expressively enriched score that includes the corresponding note onset and duration transformations and expressive performance annotations used in synthesis. The legato predictor in particular is used to guide the sample selection process in the concatenative synthesizer, described in III-B.

C. Learning of expressive performance models

For building a computational model of a set of performances of a particular performer in a particular style, we use Tilde, a top-down decision tree induction algorithm [9]. Tilde can be considered as a first order logic

extension of the C4.5 decision tree algorithm: instead of testing attribute values at the nodes of the tree, Tilde tests logical predicates. This provides the advantages of both propositional decision trees (i.e. efficiency and pruning techniques) and the use of first order logic (i.e. increased expressiveness). The increased expressiveness of first order logic not only provides a more elegant and efficient specification of the musical context of a note, but it provides a more accurate predictive model [10].

We apply the learning algorithm with target predicates `duration/3` and `energy/3`. (where `/n` at the end of the predicate name refers to the predicate arity, i.e. the number of arguments the predicate takes). Each target predicate corresponds to a particular type of transformation: `duration/3` refers to duration transformation and `energy/3` to energy transformation.

For each target predicate we use the complete training data specialized for the particular type of transformation as an example set, e.g. for `duration/3` we used the complete data set information on duration transformation (i.e. the performed duration transformation for each note in the data set). The arguments are the musical piece, the note in the piece and performed transformation.

We use (background) predicates to specify both note musical context and background information. The predicates we consider include `context/8`, `narmour/2`, `succ/2` and `member/3`. Predicate `context/8` specifies the local context of a note, i.e. its arguments are (*Note, Pitch, Dur, MetrStr, PrevPitch, PrevDur, NextPitch, NextDur*): note identifier, note’s nominal duration, duration of previous and following notes, extension of the pitch intervals between the note and the previous and following notes, and tempo at which the note is played. Predicate `narmour/2` specifies the Narmour groups to which the note belongs. Its arguments are the note identifier and a list of Narmour groups. Predicate `succ(X, Y)` means Y is the successor of X, and Predicate `member(X, L)` means X is a member of list L. Note that `succ(X, Y)` also mean X is the predecessor of Y. The `succ(X, Y)` predicate allows the specification of arbitrary-size note-context by chaining a number of successive note (4).

$$succ(X_1, X_2), succ(X_2, X_3), \dots, succ(X_{n-1}, X_n) \quad (4)$$

where X_i ($1 \leq i \leq n$) is the note of interest.

III. EXPRESSIVE PERFORMANCE SYNTHESIS

Synthesis of expressive performance in our case is the problem of rendering a score that has been previously enriched with expressivity annotations drawn from an inductive logic model by means of concatenative synthesis.

Speech synthesis research indicates, that one of the most important aspects to be handled by a convincing synthesizer are transitions between phones in addition to the phones themselves [11]. When applied to musical instrument synthesis this means that, depending on the different playing techniques supported by a particular

instrument, inter-note transitions will play an important role in the final model. In our case of jazz saxophone, we concentrated on the most notable tongued vs. legato note transition differentiation: When a note is “tongued” on a reed instrument, the air flow between mouth cavity and mouthpiece is interrupted for a short period of time by the player putting his tongue slightly below the tip of the reed. When the interruption is released again and air continues to flow, the build-up in mouth-cavity pressure causes an increase in reed excitation of the air column in the instrument. The effect of tonguing can range from a very subtle and barely noticeable alteration of the attack phase of a note to a very pronounced maximum in the energy envelope. Playing legato, on the other hand, means that the air pressure is held more or less constant over the course of a phrase and transitions in pitch to new notes are caused merely by opening or closing keys and thus shortening or lengthening the resonating air column.

As noted above, the approach taken here is to build separate models on the note level for predicting onset and energy transformations, as well as on the intra-note level, for predicting the type of transition to the next note.

A. Concatenative synthesis

The enriched, expressive score is used as input to a concatenative synthesizer [12], which, apart from depending on expressivity annotations, is completely independent from the rest of the system. The synthesizer reads an annotated input score in an extended WaveSurfer format¹.

The concatenation unit database is comprised of more than 3000 individual saxophone notes extracted from complete, expressively performed phrases. Notes are extracted and identified with their corresponding features as described in II-A.

B. Unit selection algorithm

The unit database is searched with a dynamic programming algorithm adapted from [13], [14] as described in [15] to find a sequence of database samples matching a given input score according to a cost function based on pitch, duration, timbre and musical context.

The algorithm starts by constructing a node-cost matrix with columns representing input score notes and rows denoting samples from the corpus, hereby associating a set of candidate samples $S\{t\}$ with each input score note at time t . In order to cut down the computational cost of the sample search per input score note, the sample database is divided into groups of similar samples by offline clustering, based on intra-note spectral features (currently spectral centroid, see II-A). From this clustering an additional model is created that can be applied to each input score note and annotates the note with a cluster number, which in turn determines the candidate list during sample search.

¹Additional `ATTRIBUTE:VALUE` annotations are passed on the same line as arguments to a note, delimited by spaces

At each point in time t —i.e. for each input score note—the current path cost (5) is recursively calculated for each of the corresponding candidate samples $i \in S\{t\}$, based on the current node cost $\hat{C}_i(t)$ and the previous path cost $C_j(t-1)$. Finally, the optimal path is traced from the node corresponding to the last score note that has the minimal overall path cost to the beginning of the score.

$$C_i(t) = \min_{j \in S\{t-1\}} [\hat{C}_i(t) + C_j(t-1)] \quad (5)$$

The cost function (6) is a weighted sum of two separate cost functions: The transformation cost (7) is computed from feature distance costs $F_{T_j}(t, i)$ of the score note at time t and the sample candidate i , weighted by weights w_{t_j} .

$$\hat{C}_i(t) = w_T \hat{C}_i^T(t) + w_C \hat{C}_i^C(t) \quad (6)$$

$$\hat{C}_i^T(t) = \sqrt{\sum_j (w_{T_j} F_{T_j}(t, i))^2} \quad (7)$$

The features costs used include pitch transformation cost, energy transformation cost, duration compression and expansion costs, and the transformation costs associated with the interval to the previous and the next note, respectively.

The concatenation cost (8) is composed of features based on the score note t , the sample database candidate note i and the currently accumulated best path p . The features currently used are path transition cost based on the legato descriptor, a duplicate feature penalizing reusing the same sample for different notes in the rendered phrase, a phrase membership cost rewarding the use of successive samples from the same database phrase, and a heuristic spectral continuity cost, rewarding a smooth spectral envelope for the resulting phrase based on the spectral centroid feature.

$$\hat{C}_i^C(t, p) = \sqrt{\sum_j (w_{C_j} F_{C_j}(t, i, p))^2} \quad (8)$$

Once an optimal mapping from database notes to input score notes is found, the phrase segments corresponding to notes are transformed in time and pitch by means of spectral peak processing [16], [17] and concatenated in the frequency domain in order to yield a high fidelity rendering of the expressive input score. Special care is taken to preserve attack and release segments and in particular sample transitions by only time-stretching the sustain part of the amplitude envelope.

IV. CONCLUSIONS AND FUTURE WORK

We have presented a concatenative synthesis system using dynamic programming for determining the best sequence of database samples based on a cost function derived from high-level musical features. An inductive logic model for predicting inter-note transitions in an input score was used to improve note transition quality during concatenative synthesis. Future work will concentrate on

refining transition modeling, because informal listening evaluations suggest that this is the most critical area for improvements. Another important planned undertaking is the systematic construction of a comprehensive sample database dedicated to synthesis.

ACKNOWLEDGMENTS

This work is supported by the Spanish PROSEMUS project (TIN2006-14932).

REFERENCES

- [1] G. Widmer and W. Goebel, "Computational models of expressive music performance: The state of the art," *Journal of New Music Research*, vol. 33, no. 3, pp. 203–216, 2004.
- [2] J. Kaiser and R. Schafer, "On the use of the $i0$ -sinh window for spectrum analysis," *Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing]*, *IEEE Transactions on*, vol. 28, no. 1, pp. 105–107, Feb 1980.
- [3] R. C. Maher and J. W. Beauchamp, "Fundamental frequency estimation of musical signals using a two-way mismatch procedure," *Acoustical Society of America Journal*, vol. 95, pp. 2254–2263, Apr. 1994.
- [4] R. Ramirez and A. Hazan, "Inducing a generative expressive performance model using a sequential-covering genetic algorithm," in *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2007, pp. 2159–2166.
- [5] A. Klapuri, "Sound onset detection by applying psychoacoustic knowledge," in *ICASSP '99: Proceedings of the Acoustics, Speech, and Signal Processing, 1999. on 1999 IEEE International Conference*. Washington, DC, USA: IEEE Computer Society, 1999, pp. 3089–3092.
- [6] R. McNab, L. Smith, and I. Witten, "Signal processing for melody transcription," in *Proc. 1996 Australasian Computer Science Conference*, Melbourne, Australia, January 1996, pp. 301–307.
- [7] E. Maestre and E. Gómez, "Automatic characterization of dynamics and articulation of monophonic expressive recordings," in *Proceedings of the AES 118th International Conference*, 2004.
- [8] E. Narmour, *The Analysis and Cognition of Basic Melodic Structures*. Chicago and London: The University of Chicago Press, 1990.
- [9] H. Blockeel, "Top-down induction of first order logical decision trees," Ph.D. dissertation, Department of Computer Science, Katholieke Universiteit Leuven, 1998.
- [10] R. Ramirez, A. Hazan, E. Maestre, and X. Serra, *A Machine Learning Approach to Expressive Performance in Jazz Standards*. Springer, 2006.
- [11] M. Beutnagel, A. Conkie, and A. Syrdal, "Diphone synthesis using unit selection," in *The 3rd ESCA/COCOSDA Workshop on Speech Synthesis*, Jenolan Caves, NSW, Australia, November 1998.
- [12] D. Schwartz, "Data-driven concatenative sound synthesis," Ph.D. dissertation, University of Paris 6 – Pierre et Marie Curie, 2004.
- [13] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," in *IEEE Transactions on Information Theory*. IEEE, April 1967, vol. 13, no. 2, pp. 260–269.
- [14] G. D. Forney, Jr., "The viterbi algorithm," in *Proceedings of the IEEE*, vol. 61, no. 3, March 1973, pp. 268–278.
- [15] A. Hunt and A. Black, "Unit selection in a concatenative speech synthesis system using a large speech database," *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, vol. 1, pp. 373–376 vol. 1, May 1996.
- [16] J. Laroche and M. Dolson, "Improved phase vocoder time-scale modification of audio," *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 3, pp. 323–332, May 1999.
- [17] J. Bonada, "Automatic technique in frequency domain for near-lossless time-scale modification of audio," in *Proc. 2000 International Computer Music Conference*, 2000. [Online]. Available: citeseer.ist.psu.edu/529334.html