2pMU13. Real-time concatenative synthesis for networked musical interactions

Chrisoula Alexandraki* and Rolf Bader

*Corresponding author’s address: Music Technology and Acoustics, Technological Educational Institute of Crete, Rethymnon, 74100, Crete, Greece, chrisoula@staff.teicrete.gr

The recent proliferation of Networked Music Performances has led to the investigation of low-latency, low-bitrate musical encoding schemes, including audio codecs and control protocols that specifically address the requirements of live musical interactions across the Internet. This work presents an alternative perspective inspired by the 'synthesis by analysis' approach, strictly constrained in terms of processing latencies and rendering quality. The entire process is fully automated and involves an offline processing phase (that takes place prior to performance) and a real-time analysis-synthesis phase. The offline phase involves processing a solo recording of each musician’s part so as to acquire audio segments corresponding to each note in the performance, and a trained Hidden Markov Model to be later used for online analysis. During live performance, online analysis encodes the position of the performance on a music score and resynthesizes the waveform by concatenating the audio segments of the offline phase. Although the synthesized waveform originates from an offline recording, it is synchronized to the live performance at note level, so as to allow for rendering a wide range of musical tempi as well as their expressive variations. The paper presents the complete methodology and reports on implementation details and preliminary evaluation results.

Published by the Acoustical Society of America through the American Institute of Physics
INTRODUCTION

In music communication, eliminating information bandwidth to considerably low bitrates requires the use of music descriptions instead of audio waveforms. When considering acoustic music, descriptive representations are commonly manifested by the score of a music piece or equivalently its MIDI counterpart. Although highly compact, such descriptions fail to maintain the expressive characteristics of music performed by humans. The interpretive nuances of acoustic music performances are commonly attributed to the idiomatic use of music articulation, music dynamics as well as deviations from predefined musical tempi. Moreover, the originality of sound produced by acoustic instrument manipulations is not easy to replicate when synthesizing audio from descriptive representations.

In an attempt to preserve as many of the expressive qualities of human performances as possible while keeping information bandwidth significantly low, a natural choice is to reconstruct an audio waveform by concatenating pre-recorded audio segments and doing so at a rate synchronized with a reference acoustic performance. This idea was first adopted in the domain of speech synthesis for the purposes of waveform generation in text-to-speech systems (Shoham et al. 2012; Detoit and Leich, 1993). Evidently, concatenative speech synthesis approaches yield substantial improvements in the naturalness and the intelligibility of synthesized voice compared to alternative parametric models of speech (Sak et al. 2006; Bulut et al. 2002).

The paper is structured as follows: the next section provides an overview of existing approaches to concatenative music synthesis and highlights the differences of these approaches to our target application. Then, the methodology of our approach is thoroughly delineated. The section that follows presents some preliminary experimental results that demonstrate the performance of the proposed approach. Finally, the paper concludes with a brief discussion outlining current achievements, shortcomings and future directions.

CONCATENATIVE MUSIC SYNTHESIS AND CURRENT FOCUS

In music, concatenative synthesis has attracted a lot of research interest in the last few years. For a comprehensive review of approaches, readers may refer to Schwarz (2007). The idea is to use sound snippets or ‘units’) so as to assemble a sound that matches a target prototype. The target prototype may be represented either as a sequence of sound descriptors (e.g. spectral or temporal characteristics), as a music score or MIDI stream or even as an example sound. Sound units are organized in large databases according to their characteristic descriptors and often their associated metadata (e.g. instrument class). Consequently, a substantial part of concatenative synthesis research is devoted to formulating appropriate ‘unit selection algorithms’ that can offer improved concatenation quality for the application at hand.

Like most directions in computer music research, the focus of concatenative music synthesis is either to artistically explore new sound textures or to efficiently reproduce musical acoustics. For instance, artistically explorative approaches to concatenative music synthesis are presented in the works of Aucouturier and Pachet (2006), Schwarz et al. (2006) as well as Zils and Pachet (2001). On the other hand, attempts to reproduce music signals generated by acoustic instruments are for example presented by Maestre et al. (2009) and by Lindemann (2007). The first work aims at rendering expressive saxophone performances given an arbitrary music score, while the second renders orchestral performances in response to real-time MIDI input.

In respect with online concatenation (i.e. while the input is acquired), the initiatives found in the relevant research literature are either artistically explorative or the input is provided as a description instead of an audio signal. If the target prototype is provided as an audio signal, then analysis of this signal must take place prior to matching and the whole process, including concatenation, must occur within strict time limits if input and output are to be synchronized. For example in the work of Puckette (2004), analysis and concatenation occurs in real-time in order to reproduce an output that maintains the expressive aspects of the input signal, yet is compositionally explorative and therefore not identical to the input.

In this work, the application sought is realized in the research domain of Networked Music Performance (NMP). In specific, we seek to provide a mechanism for low-latency and low-bitrate communication of musical interactions over computer networks. The focus is on realistic NMP, which refers to technologies aiming at enabling remote musical interaction that simulate co-located musical performances as accurately as possible (Alexandraki and Akoumianakis 2010). The distinguishing characteristic of such systems is that the audio latency between performers must be kept below the so called Ensemble Performance Threshold (EPT), which has been psychoacoustically measured and estimated to be of the order of 20-40ms (Schuett 2002). Consequently, the problem addressed in this work is significantly more challenging both in terms of concatenation quality as well as in terms of latencies, since
apart from the requirement that both input and output must be synchronized, the hard deadline of the EPT is imposed on the entire process of audio analysis, network transmission and re-synthesis.

In order to investigate the potential of our approach a number of assumptions must be made on the material to be analyzed and re-synthesized. For the moment the methodology has been applied to solo recordings (i.e. assuming a single musician is located at each network node) of monophonic instruments. However provisions are being made for additionally accommodating polyphonic instruments. Moreover, no unit selection process is involved, other than concatenating an automatically pre-segmented performance of the reference music piece. The entire concept might seem rather simplistic, as the segmentation of the solo performance to units, the online analysis of the live solo as well as the real-time concatenation of segments could be performed using a single algorithm for robust real-time onset detection. However, such an algorithm should provide increased accuracy in detecting note onsets as soon as they occur and before they are really perceivable. Unfortunately, the robustness of onset detection algorithms is highly unstable even more so in online and real-time settings. The methodology presented in the next section seeks to increase the robustness of real-time onset detection by progressively accumulating a trained model, able to predict onsets before they are truly detectable.

**METHODOLOGY**

The reference scenario takes into account the situation in which two or more musicians collaboratively perform the same piece of music while being physically separated and connected through a computer network. Thus each musician listens to his/her own performance in physical space, while the performance of the others is received from the network using dedicated software (such as Jacktrip or Diamouses\(^1\)). Such software would typically be responsible for transmitting the audio stream corresponding to the local performance, receiving the streams from all remote network nodes, mixing them to a single stream and rendering the mixed stream on the available audio equipment.

In the present work, we investigate the possibility of transmitting the score position of each musician instead of the audio stream. At the receiving end, score positions are mapped to audio segments and segment concatenation occurs prior to mixing. Such approach requires prior segmentation of all solo recordings, and the possibility of decoding score positions at each network site in real-time. The latter is achieved using a separate Hidden Markov Model (HMM) for the performance of each musician. The current implementation assumes that the following prerequisites hold:

a) Each instrument is located at a different networked site (i.e. waveforms correspond to solo performances)
b) Every participating instrument is monophonic (i.e. no chords are currently taken into account)
c) A solo recording for each instrument playing the piece in slow tempo is available prior to performance
d) The music score of each part is also available prior to performance

The entire methodology consists of an offline phase, that takes place prior to collaborative performance and an online phase, taking place during performance. As depicted in Figure 1, during the offline phase both the reference solo recording as well the corresponding music score (in the form of a MIDI file), are processed in order to acquire a pool of audio segments to be used for concatenation as well as a trained HMM to be used for real-time decoding of score positions. A different pool of audio segments and a different HMM is generated for each performer.

![FIGURE 1. Processes that take place offline, prior to live musical performance.](image)

During the online phase (Figure 2), the live audio stream produced at each site is analyzed using the previously trained HMM, so as to determine the current position on the reference score (i.e. the score state). If the score state

corresponds to a new note onset then a trigger is sent to the other network nodes, indicating that the next segment of the local performer should be concatenated to the currently rendered audio stream corresponding to the specific performer. In the current implementation, both offline and online processes are fully automated, i.e. no manual annotations are involved. This allows applying our methodology for arbitrary musical pieces, as long as the aforementioned prerequisites hold.

**FIGURE 2.** The online process for live networked music concatenation.

**Offline Audio Segmentation**

Segmentation is applied on the solo recording of each musician performing the piece to be played live during NMP. The possibility of segmenting the signal offline, namely when the entire waveform is available prior to segmentation, increases the robustness of any segmentation algorithm.

Assuming the waveform to be segmented is an accurate performance of the score that does not contain any wrong, skipped or inserted notes; we use blind onset detection in order to find as many onsets as the number of notes contained in the score. Commonly, onset detection algorithms comprise of three steps which are pre-processing the waveform, computation of an onset detection function (ODF), and localization of onsets at the local maxima (or minima) of the ODF (Bello et al. 2005). The ODF is usually derived as the value of some spectral or temporal feature of the signal and computed over successive blocks of audio having a pre-defined length such as 1024 or 512 samples.

After a number of empirical tests we defined our ODF as a modified version of the Spectral Flux (SF) feature. Spectral flux measures the change in spectrum among consecutive blocks of audio and it has been extensively used for onset detection in different variations, most notably in the L-1 norm used by Dixon (2006):

\[
SF_1(n) = \sum_{k=0}^{K-1} H(|X(n,k)| - |X(n-1,k)|)
\]

where \(X(n,k)\) is the k-th frequency bin of the n-th block of signal x computed using a 4096 point zero-padded FFT and \(K\) is the bin that corresponds to \(Fs/2\), i.e. \(K= 2048\). Alternatively, Bello et al. (2005) used the L-2 norm of the SF feature formulated as:

\[
SF_2(n) = \sum_{k=0}^{K-1} \{H(|X(n,k)| - |X(n-1,k)|)\}^2
\]

In both cases, \(H\) is the half-wave rectifier function: \(H(x) = \frac{x + |x|}{2}\). With rectification only the frequency bins in which the energy increases are taken into account, as this is in fact the expected behavior of spectral flux at the location of note onsets. Hence, onsets are detected as local maxima of the SF feature.
In the present work, a modified version of the Spectral Flux was used, which divides the L-1 norm with the sum of spectral magnitudes of the current audio frame, so as to prevent onsets been detected due to loud or noisy passages:

\[ SF_3(n) = \frac{\sum_{k=1}^{K} |X(n,k)| - |X(n-1,k)|}{\sum_{k=0}^{K} |X(n,k)|} \]

We found out that the SF3 feature yields increased precision of onset locations and results in very few erroneous or missed detections, even when using very small block sizes such as 256 samples per block. Figure 3 presents the value of the SF3 feature aligned on the waveform, calculated every 256 samples of a 44.1kHz/16bit drum solo recording. Onsets correspond to the top 16 maxima and they are shown on the figure as vertical green lines. The algorithm of the specific example uses two heuristic rules to reject false detections. In specific these rules reject SF3 maxima if they occur very close to each other (i.e. within a minimum inter-onset interval - IOI), as well as those occurring during silent passages. In respect with the minimum allowed IOI, the relevant literature considers an onset as accurately detected if it falls within a 50ms window of the true onset (Klapuri 1999; Duxbury Sandler Davies 2002). The computation of local maxima in Figure 3 rejects onsets that fall within a minimum IOI of 2205-samples as well as those occurring in audio blocks for which the total loudness is below the silence threshold of -70dB.

**FIGURE 3.** Blind onset detection of a drum solo using the SF3 feature calculated over consecutive 256-sample blocks.

Clearly, the signal of Figure 3 is highly percussive and therefore it exhibits strong transient regions at the location of note onsets. In practice and depending on the timbre of the instrument, it is possible to have note onsets without strong transients. Such subtle onsets can be generated when playing legato notes on wind instruments or fretless string instruments, when for example the finger slides along the fingerboard. Figure 4 shows the SF3 feature aligned on the waveform and the spectrogram for a drum solo and a flute solo recording. Strong transients are presented as vertical lines on the drum spectrogram. The flute solo does not have such strong transients. Moreover the onset occurring at approximately 0.4sec of the flute recording does not present a local maximum of the SF3 feature.

**FIGURE 4.** Salient onsets of a drum excerpt vs. subtle onsets if a flute excerpt.

In order to account for such subtle onsets, our onset detection algorithm is informed by the results of pitch tracking. Both the SF3 feature as well as pitch values are computed over successive blocks of 256 samples. The block size is deliberately small in order to provide increased time resolution in detecting onsets. Detecting pitch in
such small blocks is not generally feasible using the Fourier transform as it yields very bad frequency resolution. For this reason, we use multi-resolution analysis and in specific the Fast Lifting Wavelet Transform as presented by Maddox and Larson (2005). Pitch tracking allows for the detection of subtle onsets at the blocks where a previously constant pitch lasting for at least 150ms (i.e. three times the minimum IOI), changes to a new value and continues to have the new value for at least another 150ms. Clearly, the detection of these onsets is a non-causal process as it uses the pitch of the blocks following the potential onset.

The entire process of offline audio segmentation comprises of the following steps:

- The signal undergoes a pre-processing step which involves dc-removal and amplitude normalization.
- Feature extraction is applied in order to derive three features which are: the instant pitch (P), the normalized version of spectral flux (SF3), and the log-energy (LE)
- Subtle pitch changes are detected by examining the instant pitch. In specific a non-percussive onset is located at pitch changes for which the old pitch is maintained for at least 150ms before the potential onset and the new pitch for at least 150ms after the onset.
- The score is parsed to determine the number of notes that must be detected, for example n-notes.
- If m-notes are detected as subtle pitch onsets, then the SF3 feature is searched for the top n-m maximum values as long as they are at least 50ms apart (i.e. minimum IOI constraint) and they do not occur in blocks having the LE feature below the silence threshold of -70dB. For each SF3 maximum satisfying these constraints a new onset is recorded.
- For every onset a new segment is stored in the pool of audio segments.

### Audio-to-Score Alignment

The remaining processes of our methodology are based on an audio-to-score alignment scheme using HMMs. In specific we seek to train an HMM offline that will be subsequently used for aligning the live performance to the reference score. Every time a new note is detected the corresponding audio segment will be loaded from the pool of audio segments and concatenated to the audio stream that corresponds to the specific performer. Our approach to score following is inspired by the works of Orio and Déchelle (2001) as well as Cont (2004).

![FIGURE 5. The HMM used for real-time audio-to-score alignment.](image)

In the present work, given a score in the form of a MIDI file, we define a model that comprises of three states per note, namely attack denoted as ‘A’, sustain denoted as ‘S’, and rest denoted as ‘R’. The entire model is depicted in Figure 5. The initial state is rest followed by the three states of each of the notes contained on the score of the specific piece. Consequently, if a score comprises of n notes, the entire model comprises of N=3n+1 states. As depicted on the same figure, we define transition probabilities according to the music score. In any given state, the sum of transition probabilities is one. Non-zero probabilities are defined for allowable system transitions, which are to either remain at the same state or to proceed to the next one. If the state corresponds to sustain an additional probability of skipping the subsequent rest is defined so as to account for legato playing (i.e. no pause between successive notes).

HMM observations are derived from a number of signal features constituting feature vectors. In specific for each nth audio block of a signal denoted as x(t), if N is the length of the audio block, we compute the Log Energy feature as:

$$LE(n) = 10 \log \frac{1}{N} \sum_{t=1}^{N-1} x^2(t)$$

Apart from LE, we also use the SF3 and the Peak Structure Match (PSM) feature. The PSM feature is computed for specific frequencies corresponding to the fundamental of each pitch contained on the score. It has been extensively used in audio-to-score alignment (Orio and Dechelle 2001) and it indicates the presence of a specific pitch in an audio block. It is computed as the ratio of the energy contained in the first h bands centered on the harmonics of the fundamental frequency over the energy of the entire block. We use h=8 for the number of
harmonics and the interval of a semitone for the bandwidth of the frequency bands. For example PSM60 represents
the ratio of the energy contained in 8 semitone bands centered on the first 8 harmonics of middle C note (i.e.
261.63Hz) over the entire energy of the current audio block.

In addition to LE, SF3 and the PSM of all notes appearing on the score, we also use the differences of these
features among consecutive blocks. Consequently, the observation symbol of each audio block is the feature vector
containing LE, SF3, PSM of score notes as well as their delta values. It is important to realize that since feature
computation yields continuous values, instead of being derived from a discrete vocabulary (as in different HMM
applications), the HMM uses probability distribution functions – conventionally Gaussians – instead of an
observation matrix in order to determine the probability of each feature vector being emitted from a specific score
state.

Training the HMM

The objective of HMM training is to adapt model parameters in order to best explain a given observation
sequence (i.e. a reference recording). In particular, model parameters concern initial state, transition and observation
probabilities or probability distributions in the case of continuous observations. As we use Gaussian distributions for
emission probabilities, their mean and standard deviation (std) values could be computed using a precise manually
annotated alignment of a pre-existing recording. Experimental tests showed that with manual annotations, the
estimated mean and std values yield very accurate alignments of any given signal of the same instrument performing
the same piece of music. However, as manual annotations are not commonly available or easy to obtain, we are
looking for a fully automated training process. In this respect, the Baum-Welch algorithm (Rabiner 1989) is
proposed to iteratively converge to the correct probability values given an observation sequence (i.e. a signal). Our
tests showed that if the previously presented algorithm of offline segmentation is used to produce an approximate
alignment, then the estimated Gaussians are closer to those estimated from accurate manual annotations than the
ones computed using the Baum-Welch algorithm.

Consequently, the present methodology facilitates the offline segmentation algorithm to produce a draft
alignment, which is subsequently used to compute the relevant emission probabilities. In specific, we use the same
solo recording that was segmented, in combination with the previously computed onset times to produce an
automatically annotated dataset. This dataset is a text file in arff format2, in which, each row corresponds to a
different audio block. The row contains all feature values of the specific block followed by a class name (attack,
sustain, or rest) and a MIDI pitch value. The class name is informed by the detected onset times. Assuming the
recording starts from silence, the first blocks are tagged as ‘rest’ classes. Then, for every block corresponding to a
detected onset, the next 150ms are tagged as ‘attack’ and the blocks after the attack are tagged as ‘sustain’ up until
the block where the loudness of the signal falls below the silence threshold. When loudness gets smaller than the
silence threshold a ‘rest’ state is included. The MIDI pitches are informed from the score. Precisely, every time a
new onset is found the attack and sustain blocks are tagged with the next MIDI pitch found on the score. Rest blocks
are always tagged with zero pitch.

The resulting arff file is used to compute the mean and std values for the distribution of each feature in each state
(attack, sustain, rest). Especially, for PSM features mean and standard deviations are derived for the blocks being
tagged with the same pitch as the specific PSM feature. Then, the estimated mean and the std values are stored in a
text file and used during real-time decoding.

Real-time decoding

HMM decoding refers to the process of determining the hidden state sequence that best explains an observation
sequence given the model. In the present context, decoding translates to finding the score state for each audio block.
HMM decoding is efficiently solved using the Viterbi algorithm, which is a dynamic programming technique for
finding the optimal path among two sequences. Conventionally, it operates offline (i.e. assuming knowledge from
the entire sequences beforehand) but there are several online variants, such as the one provided in Cho and Bello
(2009) and used in the present work.

In particular, for each audio block arriving in the audio device, feature extraction is performed and an emission
probability is computed for each possible score state. This probability is based on the Gaussian distributions derived
during the training phase and it is the product of a different distribution function (probability density or cumulative
distribution function) depending on the specific feature and the score state. Specifically, for score states expected to

2 http://weka.wikispaces.com/ARFF
have high values of a specific feature (e.g. PSM at attack and sustain state) we use the cumulative distribution function, for low values (e.g. LE at rest state) we use the inverse of the cumulative distribution function, while for values concentrated around a specific range (e.g. SF3 at sustain state) we use the probability density function. This technique for computing observation probabilities has been previously proposed by Cont (2004).

Based on the observation probability computed for the upcoming audio block and the transition probability matrix depicted on Figure 5, the real-time Viterbi algorithm is applied to determine the optimal state for the current block. If the current state corresponds to new attack state then the state of the next block is examined. If it also corresponds to an attack state then a note onset is verified. As shown on Figure 2, the detection of a new onset triggers the concatenation of the next audio segment for the specific music performer at the location of remote peers.

PRELIMINARY EXPERIMENTAL RESULTS

The ideas presented in this paper have been implemented as a C++ library with different executables for the various functionalities (i.e. offline segmentation, HMM training, real-time HMM decoding and audio stream concatenation). As these ideas are still in experimental stage, the relevant software has not been fully integrated in an NMP platform so as to allow for a formal user evaluation. In this section we present some preliminary results that reveal the performance of offline segmentation and real-time HMM score-following in terms of the precision of the detected onsets as compared with manual ground truth annotations.

**TABLE 1.** Onset detection performance for offline blind detection and online HMM alignment. nGT refers to the number of ground truth onsets, CD refers to the number of correct detections, FP refers to the number of false positives (not detected onsets), and FN to the number of false negatives (falsely detected). Finally, the last column indicates the accuracy of the algorithm compared to ground truth onsets.

<table>
<thead>
<tr>
<th>Instrument</th>
<th>nGT (Blind/HMM)</th>
<th>CD (Blind/HMM)</th>
<th>FP (Blind/HMM)</th>
<th>FN (Blind/HMM)</th>
<th>Avg. Accuracy ms (Blind/HMM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flute</td>
<td>24</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>16.0/15.5</td>
</tr>
<tr>
<td>Trumpet</td>
<td>24</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>12.8/43.1</td>
</tr>
<tr>
<td>Tenor-sax</td>
<td>9</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>13.8/38.3</td>
</tr>
<tr>
<td>Guitar</td>
<td>25</td>
<td>25/19</td>
<td>2/9</td>
<td>2/3</td>
<td>11.2/16.9</td>
</tr>
<tr>
<td>Violin</td>
<td>22</td>
<td>22/20</td>
<td>0</td>
<td>0/2</td>
<td>22/14.3</td>
</tr>
</tbody>
</table>

Table 1 shows the performance of onset detection for the two techniques and for different solo monophonic instrumental recordings. It can be seen that the number of erroneous detections is very small and also that the accuracy in the detection of onsets is far below the 50ms threshold. It must be pointed out that accuracies are computed as absolute values; in fact sometimes onsets are detected before the manually annotated onset locations. Such small deviations allow for timely concatenation of the pre-recorded segments so that note progressions have a natural sounding. In our experiments for live concatenation we have used slow tempo for the solo recording available prior to performance so that segmented notes have a sufficiently large duration in order to avoid unwanted gaps between the notes.

CONCLUSIONS AND FUTURE PERSPECTIVES

In this paper a novel methodology for eliminating bandwidth demand in networked musical performances has been presented. Our pilot implementation and preliminary experimental results show that it is feasible to apply concatenative music synthesis in order to alleviate from undesirable artifacts caused by the technological constraints of currently available network infrastructures. Clearly, the current implementation is far from being a fully functional application. It rather serves as a baseline prototype that proves the feasibility of our ideas and allows for investigating further improvements addressed through continuous research and development.

Specifically, one of the main deficiencies of our present work is presented by the fact that music performers are assumed to precisely interpret the score without any errors. Clearly this is rather an ideal situation that rarely ever occurs. Our algorithms need to take into account performance errors as well as the fact that in cases where collaboration occurs for the purposes of a music rehearsal or an improvisation session, musicians will occasionally stop before the end of a music piece or repeatedly perform certain parts of the music score. We expect to address this issue by improving the HMM training algorithm to automatically learn from several audio streams, so as to be able to detect performance errors as well as arbitrary music pieces. In fact we envision a system which will be able
to progressively learn and recognize the individualities of different instruments and different performers, though continuous use.

A further improvement concerns the possibility of accommodating polyphonic and possibly multi-timbral music, therefore enabling remote music concatenation for arbitrary instruments and music pieces. We are currently investigating the possibility of incorporating chords in our model. Finally, the integration of the proposed methodology to a functional NMP platform will allow for conducting user experiments that will further inform future enhancements.

ACKNOWLEDGMENTS

This research has been co-financed by the European Union (European Social Fund – ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: THALES, Project: MusiNet

REFERENCES


