ARCHIVES OF ACOUSTICS
Vol. **42**, No. 2, pp. 235–247 (2017)

DOI: 10.1515/aoa-2017-0026

An Automatic Synthesis of Musical Phrases from Multi-Pitch Samples

Marek PLUTA[(1)], Leszek J. SPALEK[(2)], Rafał J. DELEKTA[(1)]

[(1)] *Academy of Music in Krakow*
Św. Tomasza 43, 31-027 Kraków, Poland; e-mail: psi0expikx@gmail.com

[(2)] *Institute of Physics*
Cracow Institute of Technology
Podchorążych 1, 30-084 Kraków, Poland

(*received May 4, 2016; accepted January 29, 2017*)

Sound synthesizers are a natural element of a musician's toolset. In music arrangement, samplers can often produce satisfactory results, but it requires a combination of manual and automatic methods that may be arduous at times. Concatenative Sound Synthesis reproduces many of natural performance- and expression-related nuances but at a cost of high demand for processing power. Here, another method of musical phrase synthesis aimed at music arrangement is presented that addresses these and other related issues. Contrary to common practice, we propose to record and utilize sound samples containing not one, but short sequences of pitches. In effect, natural pitch transitions are preserved and phrases appear to be much smoother, despite using very limited set of performance rules. The proof-of-concept implementation of the proposed method is discussed in detail along with attempts at optimizing the final sound effects, based on auditory tests. The limitations and future applications of the synthesizer are also discussed.

Keywords: sound synthesis; synthesis methods; sampling; Concatenative Sound Synthesis.

1. Introduction

Last few decades of advances in digital audio allowed sound synthesizers to become a standard in music arrangement. In particular, sample-based methods proved to be very efficient. They can faithfully reproduce sounds of many musical instruments, and their low complexity allows a simultaneous synthesis of large ensembles. Due to those reasons, among many synthesis methods sampling is a method of choice in music arrangement and a convenient composer's aid.

Even though not all consider sampling as a sound synthesis method, arguing that it is merely a playback of recorded sounds, others classify it in the same group of methods as wavetable and granular synthesis (Smith, 1991). All of these methods use basically the same set of signal processing techniques such as looping, resampling, filtering, amplifying, and controlling selected parameters via envelopes or oscillators. In comparison to the rest of the group, sampling stands out by using recordings of real sound sources long enough to be recognizable, and by limiting signal processing. Although its basic form is still very popular, and large libraries of high-quality sound samples are created, those high-quality samples make the limitations of the method more apparent. As a result of using samples of separate notes, synthesized note transitions in instruments such as woodwinds or strings sound unnatural, and reproduction of fluent musical passages is difficult, if not impossible. It is also difficult to control synthesized sound in a different manner than by switching to another sample, because larger amount of sample processing degrades its realism. This in turn prevents a sampler performance from being more expressive.

Where the samplers have reached their limits, other methods attempt to continue. In particular the Concatenative Sound Synthesis (CSS), inspired by a speech synthesis method (Klatt, 1983; Prudon, 2003), seems promising. It is based on sampling with elements of additive and granular synthesis. CSS uses a large database of source sounds that are segmented into units. Units are described with characteristics extracted from the source sounds or attributed to them. An algorithm finds a sequence of units that best matches the target, which is a synthesized phrase.

Units are transformed and concatenated to fully match the target (Simon *et al.*, 2005; Schwarz *et al.*, 2008).

There are various implementations of CSS. Corpus-Based Concatenative Synthesis (Schwarz *et al.*, 2006) is a content-based extension to granular synthesis. It uses grains from a large corpus of segmented and described sounds, according to proximity to a target position in the descriptor space. Units are segmented using descriptors based on the low-level MPEG-7 descriptor set, and descriptors derived from the musical score. Signal processing is mostly based on granular synthesis methods.

Reconstructive Phrase Modeling (RPM) relies on splicing fragments of phrases using additive synthesis with sine and noise components (Lindemann, 2007). One note can be built from several splices. On the basis of MIDI data RPM searches phrase database for fragments, taking into account local contexts spanning several notes. Selected fragments are stretched, pitch-shifted, time-shifted, and combined.

Expressive Concatenative Synthesis (ECS) emphasizes an expressive component of musical performance (Maestre *et al.*, 2009). It uses a set of expressive performance recordings and performance models trained using inductive logic-programming techniques. Audio data segmentation and characterization are carried out at different temporal levels (note, intra-note, note-to-note transition), and are based on fundamental frequency and energy. Processing of samples involves granular time-stretching, pitch-shifting, and energy transformation. A phrase is synthesized using phase-vocoder to concatenate samples corresponding to entire performed notes of arbitrary durations.

So far, CSS methods are much less popular in musical arrangement than sampling. They are less mature, and there are not many ready to use implementations. They are also computationally expensive, particularly during database searches, and when recordings are processed and concatenated. Recordings in the corpus do not necessarily contain parts of target phrases (e.g. in ECS the database consists of only four jazz standards played at eleven different tempos). Therefore closest matches need to be found, and then segments of selected recordings need to be pitch-shifted, time-stretched, and concatenated using various methods to avoid audible distortions. However, the more the original recordings are processed, the less naturally they sound. If pitch-shifts are not large, they can be inaudible to an untrained listener, but a musician should be able to hear the difference between natural and stretched interval due to suspiciously sounding fingering or string change.

This paper proposes another method of synthesizing musical phrases, primarily for the use in a music arrangement and as an aid for composers. It is the result of the Polish National Science Center (NCN) research project entitled "Achieving sound realism in sampling synthesis of sound of symphony orchestra wind group". The method aims at synthesizing fluent, natural phrases on the basis of musical score, while limiting signal processing and thus using less computing resources than CSS, in order to allow synthesizing large ensembles such as symphony orchestra (at present – wind instruments group). It is not intended to work in real-time, during live performance, although such expansion of the method is possible in future. Our approach shares some aspects with sampling, to keep reasonably low computational complexity, and some with CSS, to attain fluent, natural phrases. The method's original motivation, concepts, realization, an attempt at optimizing the final sound effect based on auditory tests, as well as the possible applications and its limitations are discussed.

2. The underlying concepts and their rationalization

2.1. The motivation

In the course of the work of a musician, specifically when engaged in music arrangement and composition work, the synthesis of a selection of instruments becomes necessary, as well as their mixing with the recordings of live performers. It is common practice to utilize a sampler along with a large database of samples (single note samples are standard practice) in such work. In order to achieve the most natural, realistic sound, the selection of samples is often not entirely automatized with the samples chosen manually from various databases, so as to achieve appropriately varied sounds (different articulations, dynamics etc.). Such an approach often allows one to achieve satisfactory results, however, at a cost of a great deal of work making it difficult to process large, multi-voice forms (e.g. the whole wind instrument group in the four parts of a symphony). An appropriate selection of samples is greatly dependent on the context, i.e. the music surroundings of a particular note (e.g. a chord, or a melody, or a phrase fragment). Therefore, if a suitable set of rules was to be formulated then such a task could be automated.

2.2. The initial conceptual idea

Initially, the goal was to imitate the work performed by a human during music arrangement. This included an analysis of the musical score and a choice of appropriate samples based on the details of notation (i.e. the articulation and dynamics marks, encoded in the score) as well as a knowledge with regards to the specific method of playing a particular instrument and the associated phrasing method, see Fig. 1 for a schematic representation of the initial concept.

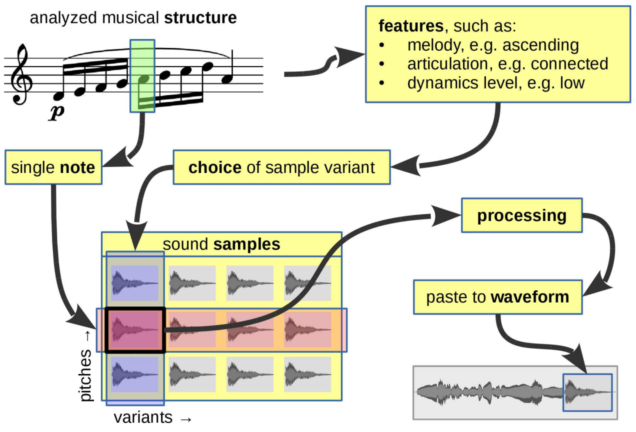

Fig. 1. A schematic representation of the initial concept of the synthesizer.

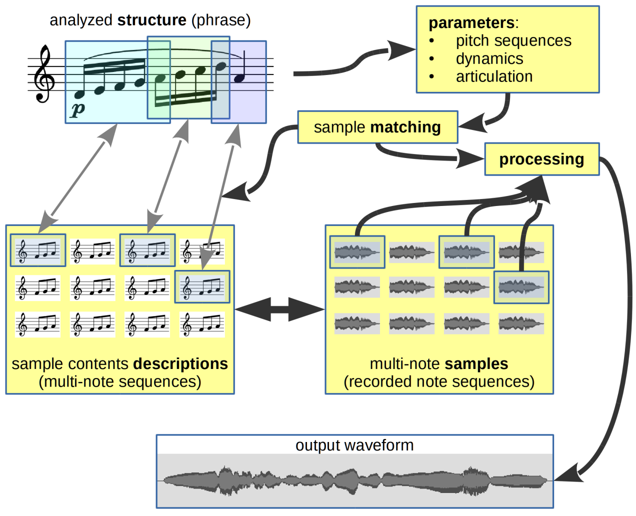

Fig. 2. A schematic representation of the current concept of the synthesizer.

Therefore, the work performed by the synthesizer was divided into two steps. The first involved an analysis of the musical score, while the second involved a synthesis of an instrument's soundtrack. In consequence, the synthesizer does not work in real-time as the analysis requires the whole of the musical score. This resembles the work of a live musician who analyzes and learns his/her own part before playing. Nevertheless, as is the case for both, a musician as well as a designed synthesizer, a particular performance of a learned/analyzed music piece may be altered in real-time to some degree. One should note that in this initial approach sound samples of single notes were being used, according to common practice.

2.3. *The current proposal*

Through our survey of the literature, we noticed that a great number of performance rules described in the literature (Friberg *et al.*, 2006; Delekta, Pluta, 2015) refers to note transitions. These are implemented unnaturally in samplers (the *legato* sequences) as the instruments part is assembled from separately registered pitches. These rules would become redundant if the recordings were to conserve the natural pitch transitions.

Therefore, our plan was to compile and record a set of short pitch sequences (containing several notes per sequence), so as to be able to assemble an arbitrary melodic line, such as those present in orchestral music from the periods of Classicism or Romanticism (see Fig. 2 for a schematic representation of this concept). Such samples would eliminate the necessity of using rules dealing with pitch transitions as the naturally performed transitions will be recorded, i.e. maintaining the appropriate parameter "variations". During the creation of a melodic sequence, adjacent multi-note samples would be concatenated on a common note (in its sustain phase). Such samples may be used in whole or in part (e.g. 3 middle notes out of the original 5 composing the whole sample). In certain areas not requiring smooth transitions, traditional samples containing single notes will be used. While small scale performance variations will be recorded in multi-pitch samples, large scale phrasing-related variations will be implemented using tempo and amplitude envelopes, according to performance rules. The original idea regarding the division into analysis and synthesis stages is retained.

In CSS methods recordings are pitch-shifted and time-stretched to match units to target. In contrast to CSS we will not use pre-recorded instrument performances of musical pieces, divided into units, as a corpus. We will utilize sample-based approach instead. With proper selection of one- and multi-pitch samples it will allow us to create any melody without pitch-shifting. In consequence, the need of complex signal processing will be limited to time-stretching only. Time-stretching is, however, unavoidable, as – contrary to pitches – it is impossible to record all note duration combinations. Our synthesizer will also need to control amplitude envelopes, but this is common for nearly all sound synthesis methods and its performance impact will be low compared to time-stretching.

3. The design and realization of the synthesizer system

3.1. *The elements of the system*

The system is composed of a set of sound samples, a set of rules, and a computer program. There are two kinds of samples: multi-pitch sequences and traditional single pitch recordings. Due to early stage of work they are currently limited to only one group of instruments. We chose the wind instruments, because here the method should bring the most benefits. Samples are accompanied by description files with information necessary for their selection and concatenation during the synthesis. The rules can be roughly divided into two categories. First category encompasses rules used to analyze a score and select matching samples. Samples are matched to the score on the basis of pitch sequences, dynamics, articulations, and approximate note durations. Rules from the second category are applied to make synthetic phrases sound more natural. They introduce parameter variations that are not present in the score, but are implied by the musical context. Finally, the software applies the rules to process the musical score and to produce output signal using selected sound samples.

3.2. *The principle of operation*

The principle of operation is schematically presented in Fig. 3. The music score serves as input data. The score is entered to the system in a TeX-like Lilypond format, which represents music in a textual form. Such score can be created and edited either directly, as a text, or by using one of several graphical music score editors.

Once entered, the score is divided into fragments that are to be played without any break ("on one breath" in case of a wind instrument), with full attack transient present only in the first note. Those fragments are further referred to as phrases, although technically they are not always equivalent to musical phrases. The division is based upon the observation that a phrase is broken by events such as pauses, note repetitions, slur ends, or large jumps in melody. The system recognizes those events and divides the score accordingly. The phrases are further divided into short segments that are matched by whole or partial sequences from multi-note samples. Up to this point, musical data is kept and processed in a textual form. Next stages of processing involve digital audio signal.

Matching samples arranged in correct order are merged into phrases. Signal processing methods and performance rules, discussed further in the text, are used to smoothly concatenate adjacent samples on a common note, and to obtain desired tempo and dynamics. A series of phrases is then composed into a final waveform (a "wav" file), which is the output of the system.

3.3. *The samples*

Two types of sounds samples are used, i.e. single and multi-note variants. The former contain various performance techniques for separated notes or special cases, such as *staccato*, accents, ornaments, notes with *crescendo* and *diminuendo*, as well as long steady notes, most of them played in two dynamic levels: *mezzo piano* and *forte*. The latter contain intervals (two-pitch sequences) from the minor second to the perfect octave, as well as tetrachords (parts of musical scales), in two dynamic levels and in two tempos (60 and 120 BPM).

For a simple reconstruction of a melody with smooth note transitions it would be enough to record interval samples only. Our selection however, was also based on the analysis of *legato* phrases from orchestral parts of wind instruments. In this kind of melodic lines it is usual that single interval "jumps" split ascending or descending scale sequences (Pluta, Delekta, 2015). Such scale sequences are rarely regular in live performances. Therefore it was important to record those irregularities in samples, and hence our inclusion of tetrachord samples.

Samples of 10 instruments were recorded in the Concert Hall of the Academy of Music in Krakow.

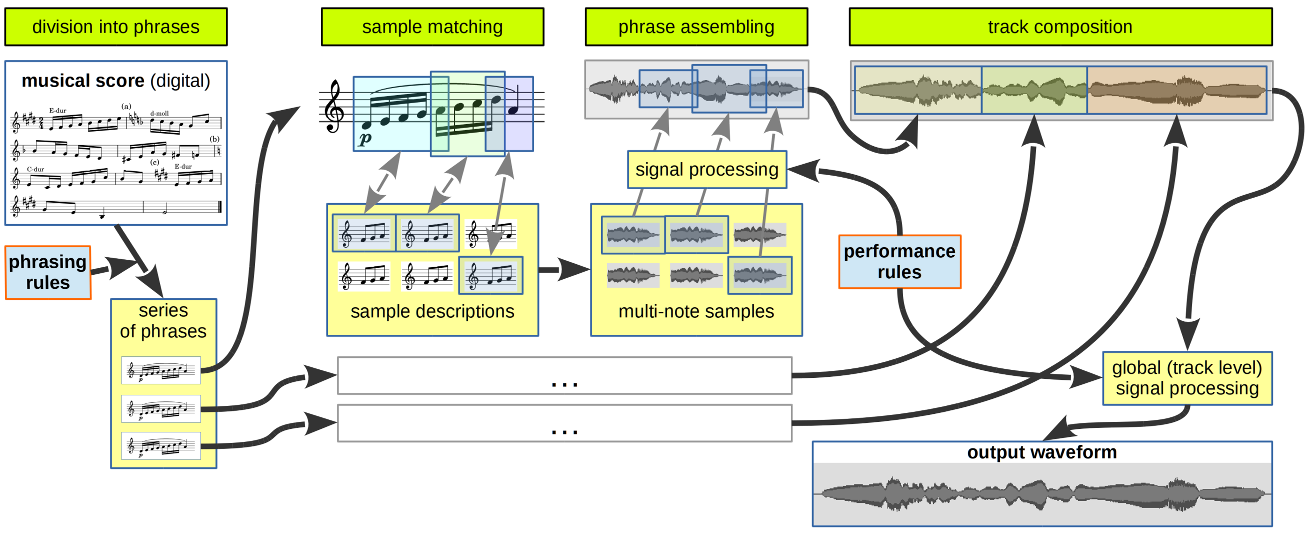

Fig. 3. The operation principle of the synthesis system. Further details may be found in Subsec. 3.2–3.5 of the text.

These include: piccolo flute, flute, oboe, English horn, clarinet, bassoon, French horn, trumpet, trombone, and tuba. Each of these instruments translates into 5000–6000 different samples. The sampling frequency was 88.2 kHz. Two stereo recording techniques were used, namely A/B with a pair of DPA 4006 microphones and X-Y with a pair of Schoeps 4V.

For each sound sample the system stores an additional textual description divided into three sections. The first one is a name of the instrument recorded. The second one – musical contents of the sample, e.g. type of recorded melodic figure, such as particular tetrachord or interval, its starting pitch, melody direction, dynamics, and tempo. It is used to match samples to phrases. The last section is needed for cutting and connecting samples in phrases. It contains segmentation data: boundaries of notes (pitches) within a sequence, represented by index of signal sample starting and ending a note. These boundaries are determined using the fundamental frequency detection algorithms with slight manual adjustments (Pluta, Delekta, 2015).

3.4. The management and selection of samples

During the final stage of score analysis the system divides phrases into short segments. These segments are to be matched with whole or partial sequences from multi-pitch samples. In order to browse the sample repository in an efficient way, the samples are stored in a fixed directory tree structure.

There are three branch levels: the instrument (root) level, the main type level, and the subtype level, where the actual samples are stored. Each of 10 currently implemented instruments has its own root-level directory that contains three main type subdirectories: single pitch samples, interval samples, and tetrachord samples. Single pitch samples consist of a number of subbranches representing various performance techniques, such as different *staccatos* (including double), long *sforzato*, long *non-vibrato*, long *crescendo*, long *diminuendo*, and ornaments (trills, mordents, and *acciaccaturas*) with four variants each (up and down, tone and semitone). Intervals are divided into subbranches according to the pitch distance (from the minor second to the perfect octave) and direction (played up or down). Similarly, the tetrachord branch contains subbranches for various types (such as lower major, upper major, and so on) and direction (up or down). All of the instruments have the same structure of subbranches.

Samples are stored in subtype subdirectories, and here they differ by the first pitch, dynamics, and tempo. Each subdirectory stores samples that cover the whole playing range of the instrument – they start with subsequent pitches. Obviously, the number of samples in a particular subtype branch depends on the instrument scale, and on the pitch structure – e.g. there are less samples with larger intervals than with smaller. Apart from different pitches, where applicable, samples are recorded in two dynamics levels (*mezzo piano* and *forte*), and in case of multi-pitch samples – in two tempo variants (60 and 120 BPM). Additionally, the most common samples (such as minor and major seconds) are recorded in several variants to be used interchangeably and thus avoid repeatability. Each sample parameter, that is the first pitch, dynamics, tempo, and variant, are encoded into a unique filename. The name of a sample file and its position in the tree fully describes its content. When the system requests a particular sample, it simply creates a proper filename and path according to a fixed set of rules.

A sample matching is a process in which a phrase is recreated using overlapping sequences of pitches from the available samples (Pluta, Delekta, 2015). Whole or partial pitch sequences from the samples can be used. The only limitation is the beginning of the phrase. It has to start with a beginning of the sample to recreate the attack transient. The following samples can be cut as needed. Apart from pitches, also dynamics, tempo, and articulation are considered. Firstly, the system finds all possible sample sequences that match the phrase. Secondly, from all the candidates, the sequence with a smallest number of required concatenations is selected. In a rare case when a part of phrase cannot be matched by any available multi-pitch sample, this section of phrase is recreated as in standard samplers – with single-pitch samples.

The current set of samples is complete for the wind instruments group. However, the structure and consequently the matching algorithm will need to be adjusted in case of implementing another group of instruments, such as strings, with different playing techniques.

3.5. Performance rules

The aim of implementing performance rules is a desire to reduce the gap between the output of a sound synthesizer and the recording of live musician. The latter are distinct through the presence of minute variations in note durations, fine tuning of pitch, articulation, dynamics, and note connections. Variations of this kind are not a part of the musical score, and are not present in a literal, regular synthetic reproduction. They are the result of a human performer's way of thinking about the melody: its tension and climaxes, as well as its context and aims. Obviously, they vary from performer to performer. A vast amount of research was carried out in order to establish a quantitative description of this phenomenon (Bresin, 1998; Friberg *et al.*, 2006; Gabrielsson, 1985; Widmer, 1995; 2002; Widmer, Tobudic, 2003), which resulted in a number of universal "performance rules". These rules were applied in a number of sequencers to mimic human performance while controlling synthesizers.

Performance rules match specific melodic, harmonic or rhythmic structures (a context of a note) with variations of quantities that characterize the performance. There are several groups of rules, such as phrasing, micro-level timing, tension, intonation, synchronization, performance noise, and basic articulations (Bresin *et al.*, 2002; Delekta, Pluta, 2015). In case of synthesizers which generate individual pitches separately, the rules are applied to the following parameters: signal amplitude, inter-onset duration (note spacing), offset to onset duration (note duration), vibrato amplitude, and deviation from 12-TET tuning (Friberg *et al.*, 2006). Our system synthesizes musical phrases using recorded sequences of pitches that already contain some variations, e.g. related to ascending and descending melody in tetrachords, or resulting from note transitions in intervals. It makes a number of performance rules obsolete. However, signal processing routines used to modify recorded sequences require additional control. Hence the application of rules involves not only pitch fine-tuning, signal amplitude, or note duration, but also amplitude and tempo envelopes, as well as sample selection.

Rules selected for our system are based on the KTH set (Bresin *et al.*, 2002) and are presented in Table 1. The phrasing group is particularly important with rules such as the phrase arch, the final *ritardando*, and the melody accent. The application of rules involves some user interaction. For example in case of phrase arch user sets nodes for tempo and amplitude, or modifies nodes set automatically by the synthesizer on the basis of slurs and dynamics markings in the score. Having established the nodes, the system calculates envelopes by application of shape-preserving interpolation with smooth first derivative using piece-wise cubic Hermite interpolating polynomial. The envelopes control variations of appropriate parameters.

Table 1. A selection of performance rules that can be applied in our synthesizer, based on the KTH set (Bresin *et al.*, 2002). Parameters are: pitch fine-tuning (PT), signal amplitude (SA), note duration (ND), amplitude envelope (AE), tempo envelope (TE), and sample selection (SS).

Rule name	Comment	PT	SA	ND	AE	TE	SS
Phrasing							
Breath	Small pauses between phrases			+			+
Phrase arch	Arch-like change in tempo and signal amplitude				+	+	
Final *ritardando*	Slow down in the end of a piece					+	
High loud	Signal amplitude proportional to pitch				+		
Metric accent	Emphasize metrical structure				+	+	
Melody accent	Emphasize melody climax				+	+	
Micro-level timing							
Duration contrast	Shorten short and lengthen long notes					+	
Faster up	Increase tempo in ascending pitch sequence					+	
Tension							
Melodic charge	Emphasize distance from current chord		+				
Harmonic charge	Emphasize distance from current key		+				
Chromatic charge	Emphasize sequences with chromatic changes				+		
Intonation							
High sharp	Stretch intervals according to size	+					
Mixed intonation	Tuning to harmonic context in long chords, and to melodic context in fast sequences	+					
Synchronization							
Melodic sync	Synchronize notes in each voice to the “collective voice” (with notes from all voices)					+	
Performance noise							
Noise control	Introduction of human-like inaccuracies				+	+	+
Basic articulations							
Legato and *staccato*	Two basic changes from normal articulation						+
Repetition	Slight separation of repeated notes			+			+

3.6. The software

The system is composed of 4 modules, see Fig. 4 for a schematic overview after (Pluta *et al.*, 2016). The preliminary analysis is carried out by the score analysis module. A selection of samples that will constitute components of musical phrases is carried out next via the figure matching module. A wav file is created through sample cutting, joining and processing performed by the waveform generator. The synthesis processes are coordinated by the management module by launching the respective modules in appropriate order, overseeing the data flow, as well as coordinating and synchronizing the parameters of voices in the case of polyphonic synthesis.

User interaction is necessary during the first stage of the synthesis process, and is possible in further stages. Initially, the user provides a digital musical score (see Subsec. 3.2), chooses which performance rules should be enabled during synthesis (see Subsec. 3.5), and assigns voices in case of polyphony. Later, the management module allows the user to modify or correct output of the score analysis module and the figure matching module. During this stage the user can enter data for selected performance rules, such as additional nodes for phrase arches, or correct automatically assigned values. In current implementation all user interaction is performed through edition of configuration files. Such design allows either further addition of a graphical front-end to user actions, or substituting the user with higher level control software.

Modular processing and storing data from all the stages of the synthesis process allows to produce different performances of the same piece of music by returning to a certain stage and changing selected parameters. It is possible to modify the waveform generator to add a layer providing real-time control over a performance on the basis of previous score analysis, e.g. control over signal amplitude and tempo for orchestra conductors training.

3.7. The implementation

The system has been implemented as a proof-of-concept of the proposed synthesis method. It aims to be open for modifications and future expansions via additional modules, such as real time control over performance, or global control over all performance rules according to a chosen performance style. Therefore the implementation is not user-oriented. Instead it is controlled through textual configuration files, which allows it to interact with both, human user or other programs.

GNU Octave was selected as the environment of choice due to the open source nature of the software, its comprehensive documentation, the concise and clear nature of the produced code, a great number of ready to use functions including those associated with signal processing as well as its easy integration with other open source tools.

The synthesizer software is in the form of a series of Octave scripts, which supports multiplatform interoperability, i.e. with the capability of running on various computer platforms including Linux, Microsoft Windows, MacOS and others that may run Octave. There are main scripts that represent modules of the system, as well as supplementary scripts that contain implementations of particular data or signal processing algorithms, such as sample matching, modified cross-fade or shaping the amplitude envelope.

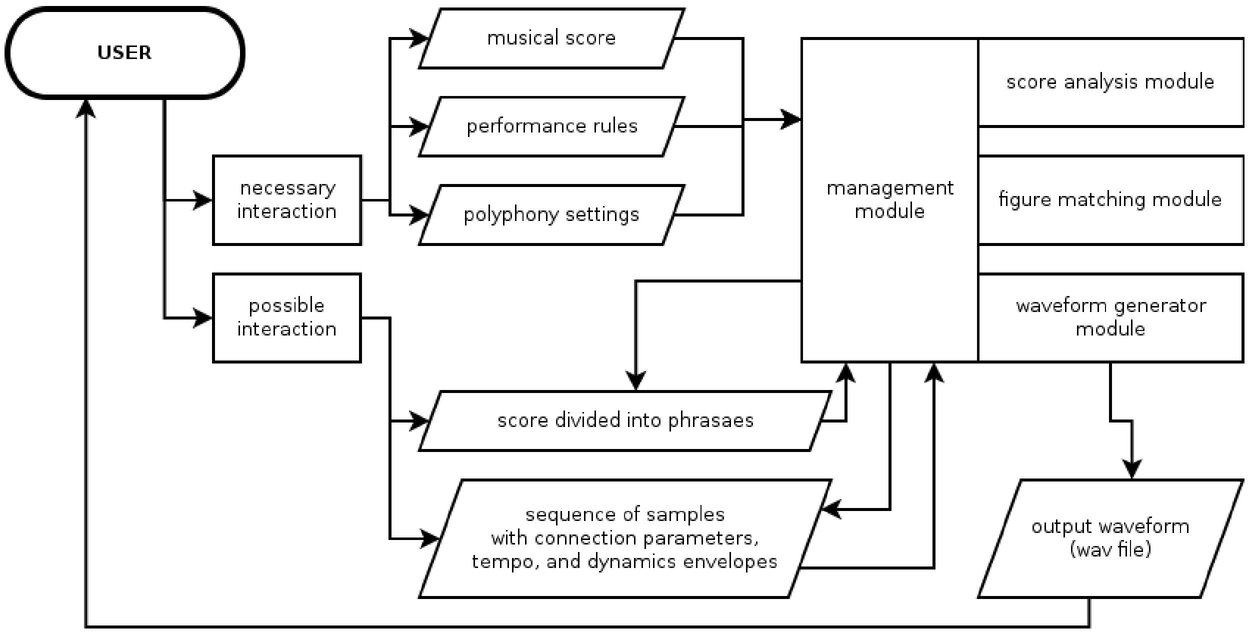
USER
necessary interaction
possible interaction
musical score
performance rules
polyphony settings
management module
score analysis module
figure matching module
waveform generator module
score divided into phrasaes
sequence of samples with connection parameters, tempo, and dynamics envelopes
output waveform (wav file)

Fig. 4. A schematic representation of the system modules, after (Pluta *et al.*, 2016).

4. Details of operation

4.1. Sample concatenation

In the process of musical phrase synthesis fragments of multi-note samples have to be assembled into a smooth melody. In order to keep natural pitch transitions intact, the synthesizer cuts the appropriate fragments of adjacent samples and performs the concatenation. Unlike in traditional sampling, where single-pitch samples are connected between pitches and natural transitions are lost, here the connection is performed by overlapping and cross-fading a sustain phase of a common pitch of two multi-note samples.

Both connected samples are highly correlated in the cross-fade region – they have the same timbre and pitch. Therefore, depending on both signals phases, regular cross-fade could lead to momentary audible decreases in amplitude, and cannot be directly applied. In order to remedy this problem a modified cross-fade method with phase alignment was developed and applied, cf. Fig. 5. The second sample is time-shifted in relation to the first one, so that both are in phase. The phase adjustment value is calculated as an abscissa of first maximum of a cross-correlation between both samples in overlapping region.

In most cases the shift needed to align the phases was observed to be smaller than 1 ms. Even in extreme cases it only approached 10 ms, and it was the worst case scenario (half-period shift) for a pitch with 50 Hz fundamental frequency. Such values are acceptable in comparison to 100 ms – a value considered to be the fastest perceptual musical separation possible (London, 2004).

4.2. Control of the rhythm and tempo

Multi-note samples have their own tempo, different from that of the synthesized fragment. When performing a synthesis the tempo of the samples needs to be adjusted to the tempo of the target piece. This is done through note extending and shortening, see Fig. 6. Note is shortened by cross-fading it with its time-shifted copy (shift value is negative), which re-

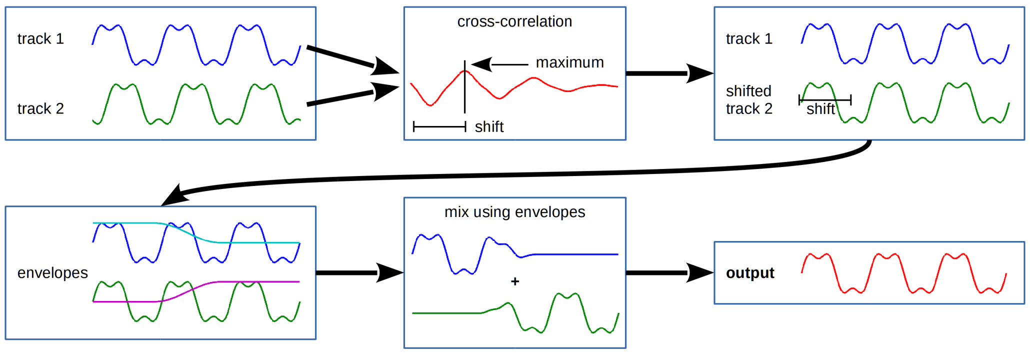

Fig. 5. A schematic representation of the modified cross-fade algorithm implementation.

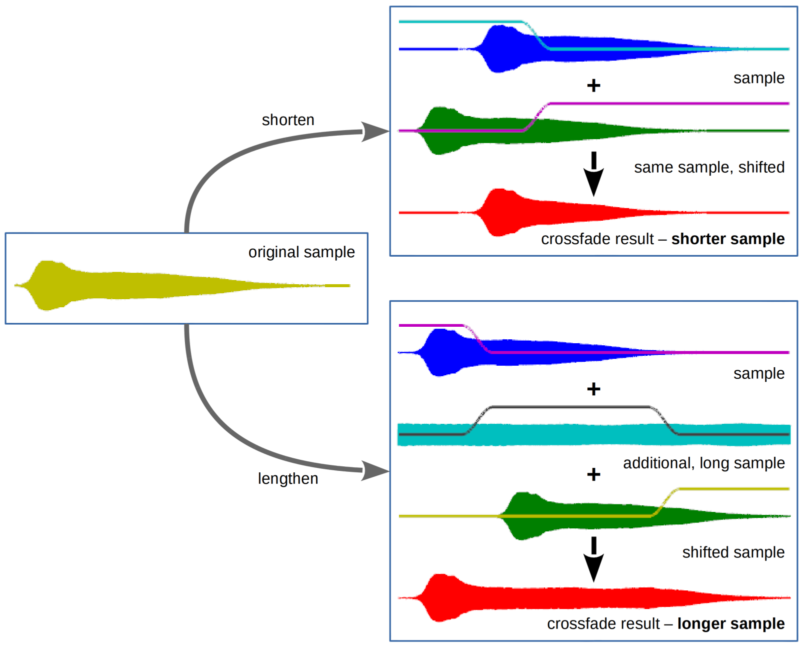

Fig. 6. A schematic representation of the note duration change implementation.

moves superfluous part of the sustain region. Note extension requires two cross-fade operations and an additional sample with a single long note. Beginning of the extended note is cross-faded with a long note, and long note is cross-faded with time-shifted ending of the extended note (shift value is positive), which inserts additional sustain region. A rhythm is created by picking samples with note durations closest to the target durations, and adjusting them as needed by note extending and shortening.

Whenever the performance rules require applying a tempo envelope, the time locations of the individual notes in the final track are determined according to the method schematically presented in Fig. 7. We start by determining an initial, regular tempo lattice. We then modify this lattice according to the envelope, following the performance rules. We finally fit in samples into the locations determined by the aforementioned lattice by performing appropriate cuts and insertions (as described in the previous section).

4.3. Control of the dynamics

Dynamics in music affects not only a signal level, but also a performance technique. To recreate it, our system handles dynamics in two ways. Firstly, it uses separate sound samples for different dynamic levels (we use two recorded levels: *mezzo piano* and *forte*). These are chosen according to the musical score. Secondly, an amplitude envelope is applied to the synthesized signal to shape finer variations according to both: the score and the performance rules. The envelope is applied in two steps. The first one is note-to-note level normalization. It is necessary due to different sound levels within each sample. This is carried out by calculating the RMS value of each individual note and then by applying the resultant amplification coefficients. In effect, the phrase is synthesized with globally flat music dynamics, and only local fluctuations. In the second step the amplitude envelope is applied over the flat dynamics. The overall process is shown in Fig. 8.

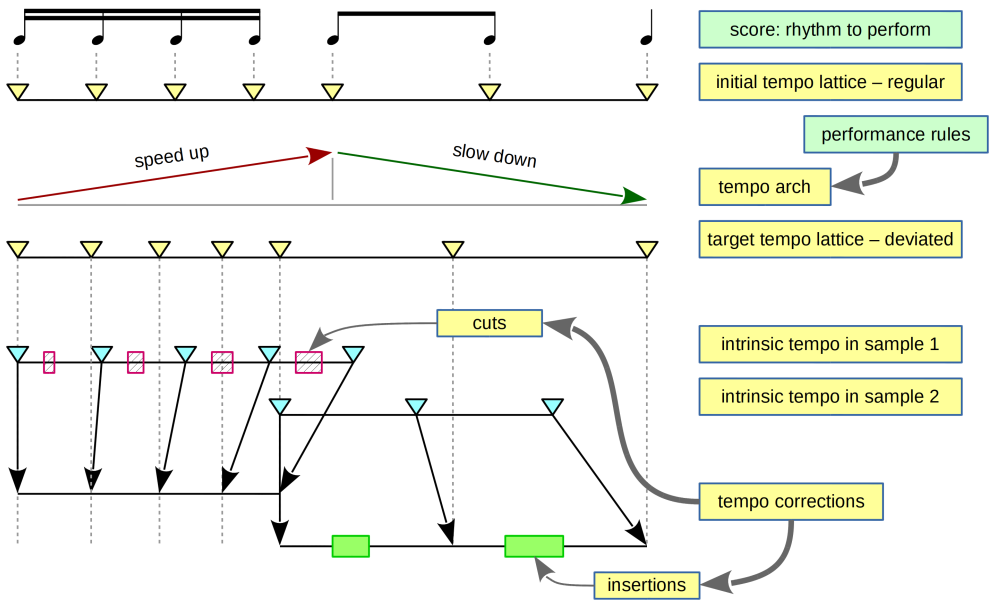
score: rhythm to perform
initial tempo lattice – regular
performance rules
speed up
slow down
tempo arch
target tempo lattice – deviated
cuts
intrinsic tempo in sample 1
intrinsic tempo in sample 2
tempo corrections
insertions

Fig. 7. A schematic representation of the method of sample tempo adjustment to the target tempo, including the application of the tempo arch (tempo envelope control).

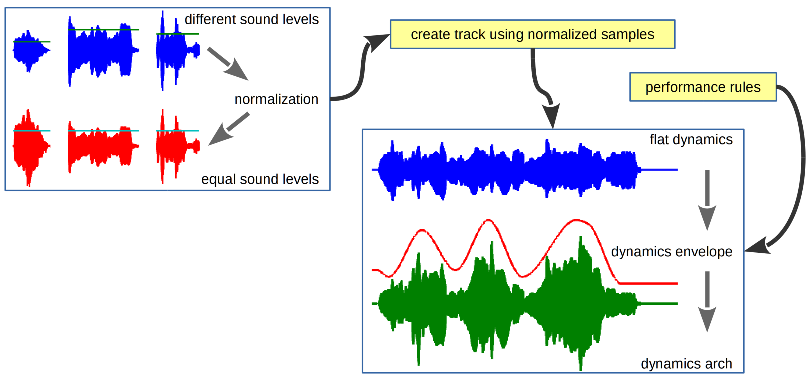

Fig. 8. A schematic representation of the sample level equalization and the implementation of the dynamics envelope.

5. An adjustment of the parameters of the system

5.1. Phase I

The very first working implementation of the synthesis system produced mixed results. Some fragments sounded natural, but there were also numerous problems. For instance, introducing effects deemed to positively impact the result (like the phrase arch) involved more signal processing, which in turn frequently produced audible distortions. Obviously the algorithms required tuning. We attempted to isolate the algorithms and parameters that affected the result the most.

The initial listening test (Delekta *et al.*, 2016) was carried out with the aim of comparing and evaluating variants of musical performances produced by the synthesis system with different sets of parameters. Specifically, the listeners compared pairs of sound sample variants based on flute and bassoon parts from symphonies by A. Dvořák and W.A. Mozart, respectively. Listeners were instructed to make a single selection based on their perception of the best resemblance to natural sound transitions. The setting of the experiment is presented in Table 2.

Table 2. The setting of the experiment – phase I.

Test participants	15 listeners: 10 experienced university-level ear training teachers, and 5 PhD student orchestra conductors.
Procedure	Fixed order of pairs of samples presented. Forced choice of one sample from each pair. Playback and repetitions on demand. Procedure controlled automatically by a computer software.
Listening conditions	Closed studio headphones (Beyerdynamic DT 770 Pro). Sound level set individually per listener. Listeners allowed to take breaks at any moment during the test.
Signal presentation	Diotic (1–channel/mono samples, presented simultaneously to each ear).
Test duration	50–120 minutes (in most cases approximately 70 minutes).

The test yielded detailed results concerning the specifics of the algorithm's implementations such as the length of cross-fade region and areas of the note excluded from cross-fading (beginning and ending). Cross-fades between 60 and 120 ms were preferred. Short cross-fades (below 30 ms) were particularly badly received. Among several tempo variants of the same piece, the listeners preferred variants where the tempo was closest to the intrinsic tempo of samples. It hints towards the use of samples with a greater variety of tempos in future implementations. The listeners did not show any statistically significant preference towards any of the phrasal variants, i.e. with or without dynamics or tempo changes. While this may seem surprising, according to further inquiry the listeners could tell the difference between the samples, but the choice was a matter of preference towards more flat, or more expressive performance. Listeners claimed that both variants were flawed: flat was obviously unnatural, but the alternative expressive variant was also not convincing.

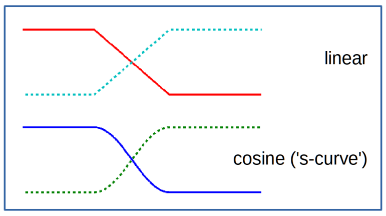

Fig. 9. A comparison of a linear and cosine envelope. The latter was used in the newer version of the algorithm for realizing cross-fade, fade-in, and fade-out segments. The linear variant was used in all of the previous cases (older version).

Listeners were additionally asked for feedback which was subsequently used for further improvements of the synthesizer system parameters, see Subsec. 5.2. They noted that longer notes appeared idle, in contrast to how they are normally performed. Remarks were also made regarding the silence of the recording room, which should be audible in the track background during the whole duration of the playback as complete silence during rests was perceived as unnatural. There was also a considerable number of remarks with regards to articulation and accentuation, however, these were mutually contradictory.

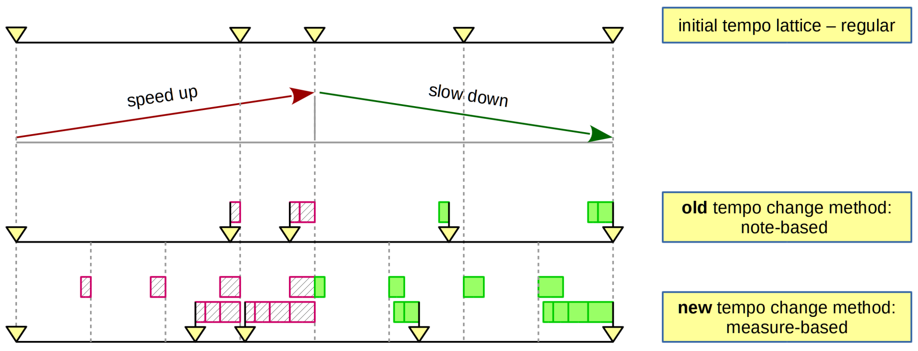
initial tempo lattice – regular
speed up
slow down
old tempo change method: note-based
new tempo change method: measure-based

Fig. 10. A comparison of the old and new tempo phrase arch implementation.

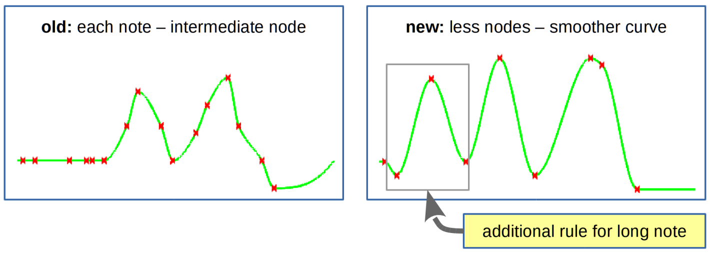

Fig. 11. A comparison of the old and new dynamics envelope.

5.2. Phase II

The second phase represents a second iteration of parameter optimizations. These constitute an improvement over the results obtained during the first stage as modifications have been implemented based on the opinions of the test subjects expressed in that phase. The details are discussed below.

5.2.1. Test details

A fragment of the flute part from Symphony No. 5 in F major, Op. 76, III Movement – Trio, bars 285–292 by A. Dvořák was used in the listening tests. This particular piece was chosen so as to determine the influence of parameter values on closely combined sounds in a melodious phrase.

The test group consisted of 23 listeners. The composition of the group was varied and included violin, composition, and conducting students from the Academy of Music in Krakow, engineers and acousticians as well as ear training teachers from the Academy of Music in Krakow.

Speakers (near-field studio monitors) or closed studio headphones were used for diotic signal presentation. The varied nature of the test conditions was thus used to resemble real listening conditions, as the system is to be used in various settings.

Test samples were grouped into 7 sets (from A to G). Set A consisted of 4 samples (further referred to as variants), and the rest (B–G) consisted of 2 variants each. Variants in sets A–F differed in only one aspect, as described in Table 3, addressing problems signaled

Table 3. Sample sets used in test phase II. Keyword "old" in column "Variants" marks a variant from phase I, as a baseline for comparison.

Set	Differences between samples within a set	Variants within a sample set
A	Background noise presence and level. Noise was registered in a recording studio. Noise level is relative to the signal level, RMS.	1: no background noise (old) 2: noise level −45 dB 3: noise level −39 dB 4: noise level −33 dB
B	Envelopes of fade-in, fade-out, and cross-fade segments (cf. Fig. 9).	1: cosine ("s-curve") 2: linear (old)
C	Duration of fade-in and fade-out segments.	1: fade-in/out 20/80 ms (old) 2: 120/60 ms (softer onsets)
D	Presence of the tempo phrase arch (cf. Fig. 10 – the new method). Compared to the method used in phase I, the new one corresponds to smaller (up to 3%) and smoother tempo changes. A continuous curve is calculated per unit value of rhythm (old was per note) on the basis of tempo values given at nodes such as culminations or endings.	1: tempo arch present 2: no tempo variations
E	Presence of the dynamics (sound level) phrase arch (cf. Fig. 11 – the new envelope). Compared to the dynamics envelope used in phase I, the new one is smoother. It is calculated on the basis of values given at fewer nodes (old was one node per note). It also adds emphasis on long notes.	1: dynamics arch present 2: no variations in dynamics
F	Fine tuning of pitches to 12-TET system. Usage of samples without tuning results in minute, however audible discontinuities in pitch on some cross-fades. It is corrected by 12-TET tuning, but averaged pitches are not perceived very well by the listeners under some circumstances (e.g. the leading notes or some of the interval jumps).	1: pitches tuned to 12-TET 2: originally recorded tuning (old)
G	All improvements from A to F together. All the improvements introduced in A–F (A2 B1 C2 D1 E1 F1) were implemented together and compared as a whole with the old variant (A1 B2 C1, phrase arches D and E in the form from phase I, F2).	1: all improvements together 2: old (as in phase I)

by the listeners in previous phase. All the modifications implemented in sets A–F were put together in set G. The listeners were given a task of indicating better sounding variant in each set. The number of playback repetitions was not restricted. Not all of the trials contain equal numbers. In a few single cases the listeners did not hear any differences and did not provide an answer (once in the case of B and once in the case of F).

5.2.2. Results

The results have been gathered in Fig. 12 below with accompanying commentary. Due to relatively small number of listeners (23) the results cannot be deemed statistically significant at a confidence level of 0.95 (set G constitutes a borderline case). At a confidence level of 0.90 the difference in set G becomes statistically significant and exhibits a preference towards the new version, while differences in other sets remain insignificant albeit with a weak preference towards the newer in set E (dynamic arch) and F (tuning). In A (background noise) the preference would be statistically significant in favor of the new variant (noise) if all variants with noise present (2–4) were counted together and treated as a "no-noise vs noise" case.

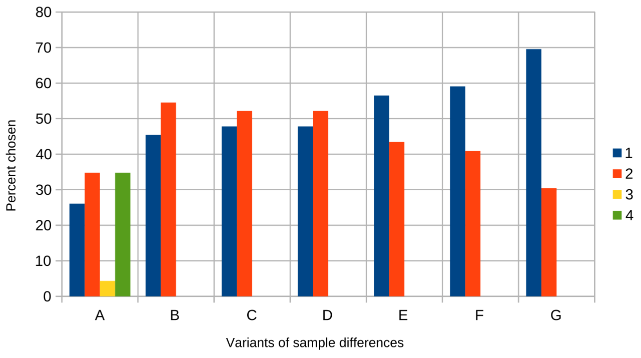

Fig. 12. A compilation of the results obtained in the second phase of the system adjustments. Letters A–G represent sample sets (as described in Table 3), and numbers 1–4 represent sample variants within a set (4 variants in set A, 2 variants in B–G). Please see the text for details of the listening test.

5.2.3. A discussion of the results obtained

Most of the differences between variants in phase II were considered "almost inaudible" by the listeners in the inquiry after the test. It is understandable particularly in cases such as cross-fade envelope or a very subtle new tempo arch. Nevertheless, all the modifications were tested separately to find out whether any of them is particularly important.

As there are no clear preferences in sets B–D, these settings are to be left as a parameter of choice for the user. The very weak (and statistically insignificant) improvement in set E (the dynamic arch) in comparison to the results obtained in the first phase may be caused by the introduction of the additional changes in the long notes. Despite an insignificant result in the case of set F (intonation), the preference is towards the tuned version. These weak preferences might become statistically significant if a much larger group were to be tested.

Set G appears to be the most interesting case. Despite the fact that on their own, none of the improved versions (A–F) was clearly better judged, a simultaneous application of all of the corrections is perceived better than the older set. This version will be now taken as the default with the possibility of making parameter changes by the system user to the values of the previous set according to individual preferences.

5.3. Phase III

Although the system is still a work in progress, we carried out a small comparison with a commercial sampler: Independence Premium Library (70 GB version) included in Samplitude Pro X Suite DAW. Setting of the experiment was the same as in phase II, but the group of listeners was smaller (17 people – musicians, acousticians, and sound engineers). We used the same flute fragment from Dvořák Symphony No. 5. The listeners compared two variants of the piece, one created by our synthesizer and the other from the commercial sampler, with a task of selecting a better sounding one. All settings of our synthesizer were exactly the same as in variant G1 from phase II. We entered score to Independence in the form of a standard MIDI file.

Out of 17 listeners 8 have chosen our synthesizer and 9 have chosen the sampler. Considering the fact that we compared a mature, refined, commercial solution to experimental implementation of our method, the results are encouraging with similar groups preferring either solution. On the other hand it is clear that the method needs refining. In comparison to samplers it has more limitations and uses more computing resources, so it needs to produce better results to become a viable alternative.

6. Conclusions

A method for automatic synthesis of musical phrases was formulated, and its proof-of-concept implementation was presented. The method provides a new tool for the use in music arrangement that aims at producing more natural musical phrases than samplers without complex signal processing of CSS methods. It synthesizes phrases from multi-pitch samples to keep natural note transitions and applies performance rules to simulate musical expression. Unlike CSS methods that require pitch-shifting and time-stretching to match recorded note sequences to target phrases, our method uses samples with all possible note transitions, thus requiring only time-stretching algorithm.

The main drawback of the presented method is a requirement for a very large sound sample database.

Therefore, a potential commercial version will require costly albeit one-time recordings and time consuming sample processing. Another problem is the sample processing itself, and particularly segmentation of pitches and sustain phases in samples. Current methods based on fundamental frequency detection with manual corrections proved both time-consuming and inaccurate. A more elaborate solution needs to be developed since, as for now, imprecisely segmented samples cause audible distortions in some situations. We plan to utilize CSS segmentation methods with frequency-based and amplitude-based audio descriptors as well as tonal/noise distinction.

The system is not intended for real-time or live performance playback. Its main application area lies in the music arrangement and related domains, i.e. whenever the music score is known beforehand and may be analyzed prior to synthesis. Work in a quasi real-time mode may be envisaged. An analysis of the score would be performed first, followed by a real-time control or adjustment of performance parameters, such as dynamics and tempo.

Acknowledgments

This study is a part of the 2012/05/B/HS2/03972 research project supported by the Polish National Science Centre.

References

1. Bresin R. (1998), *Artificial neural networks based models for automatic performance of musical scores*, Journal of New Music Research, **27**, 239–270.
2. Bresin R., Friberg A., Sundberg J. (2002), *Director Musices: The KTH Performance Rules System*, Proceedings of SIGMUS-46, pp. 43–48, Kyoto.
3. Czerwiński A., Grzybek D., Krauze P., Łuczko J., Michalczyk K., Orkisz P., Pluta M., Radziszewski L., Saga M., Snamina J. (2015), *GPU-based sound synthesis controlled with MIDI protocol* [in Polish: *Synteza dźwięku z wykorzystaniem procesora graficznego, sterowana przy użyciu protokołu MIDI*], [in:] *Selected issues of noise and vibration reduction systems* [in Polish: *Wybrane zagadnienia układów redukcji drgań i hałasu*], Orkisz, P. [Ed.], pp. 40–52, Akademia Górniczo-Hutnicza, Kraków.
4. Delekta R.J., Pluta M. (2015), *An implementation of the performance rules in the modified sampling synthesis of wind instruments* [in Polish: *Implementacja reguł wykonawczych w syntezie dźwięku instrumentów dętych zmodyfikowaną metodą samplingową*], Proceedings of XVI International Symposium on Sound Engineering and Tonmeistering ISSET'2015, pp. 119–125, Warszawa.
5. Delekta R.J., Spalek L.J., Pluta M. (2016), *The impact of selected parameters of a modified sampling synthesis on the result of its auditory assessment*, Journal of Applied Mathematics and Physics, **4**, 2, 221–226.
6. Friberg A., Bresin R., Sundberg J. (2006), *Overview of the KTH rule system for music performance*, Advances in Cognitive Psychology, **2**, 2–3, 145–161.
7. Gabrielsson A. (1985), *Interplay between analysis and synthesis in studies of music performance and music experience*, Psychology of Music, **3**, 59–86.
8. Klatt D.H. (1983), *Review of Text-to-Speech Conversion for English*, Journal of the Acoustics Society of America, **82** 3, 737–793.
9. Lindemann E. (2007), *Music Synthesis with Reconstructive Phrase Modeling.* IEEE Signal Processing Magazine, **24**, 2, 80–91.
10. London J. (2004), *Hearing in Time: Psychological Aspects of Musical Meter*, Oxford University Press, Oxford.
11. Maestre E., Ramirez R., Kersten S., Serra X. (2009), *Expressive concatenative synthesis by reusing samples from real performance recordings*, Computer Music Journal, **33**, 4, 23–42.
12. Pluta M., Delekta, R.J. (2015), *Technique to Seamlessly Connect Sound Samples in Sampling Synthesis*, [in:] Progress of Acoustics 2015, Opieliński K.J. [Ed.], pp. 271–282, Polskie Towarzystwo Akustyczne, Wrocław.
13. Pluta M., Spalek L.J., Delekta R.J. (2016), *A modified sampling synthesis for a realistic simulation of wind instruments – the design and implementation*, Journal of Applied Mathematics and Physics, **4**, 2, 215–220.
14. Prudon R. (2003), *A Selection/Concatenation TTS Synthesis System*, PhD Thesis, LIMSI, University of Paris XI.
15. Schwarz D., Beller G., Verbrugghe B., Britton S. (2006), *Real-Time Corpus-Based Concatenative Synthesis with CataRT*, Proceedings of the COST-G6 Conference on Digital Audio Effects (DAFx), pp. 279–282, Montreal.
16. Schwarz D., Cahen R., Britton S. (2008), *Principles and applications of interactive corpus-based concatenative synthesis*, Proceedings of the Journées d'Informatique Musicale (JIM).
17. Simon I., Basu S., Salesin D., Agrawala M. (2005), *Audio Analogies: Creating New Music from an Existing Performance by Concatenative Synthesis*, Proceedings of the 2005 International Computer Music Conference, pp. 65–72, San Francisco.
18. Smith III J.O. (1991), *Viewpoints on the History of Digital Synthesis*, Proceedings of the International Computer Music Conference ICMC-91, pp. 1–10, Montreal.
19. Widmer G. (1995), *Modeling rational basis for musical expression* Computer Music Journal, **19**, 76–96.
20. Widmer G. (2002), *Machine discoveries: A few simple, robust local expression principles*, Journal of New Music Research, **31**, 37–50.
21. Widmer G., Tobudic A. (2003), *Playing Mozart by analogy: Learning multi-level timing and dynamics strategies*, Journal of New Music Research, **32**, 259–268.