Real-Time Sound Similarity Synthesis by an Ahead-of-Time Feature Extraction

Marco Matteo Markidis
nonoLab
V.le Vittoria 3
Parma, Italy, 43125
mm.markidis@gmail.com

José Miguel Fernández
IRCAM
Place Igor-Stravinsky 1
Paris, France, 75004
jose.miguel.fernandez@ircam.fr

Giuseppe Silvi
Centro Ricerche Musicali
Via Giacomo Peroni 452
Rome, Italy, 00131
me@giuseppesilvi.com

Abstract

This paper presents *path~*, a Pure Data external implementing a corpus-based, concatenative analysis-synthesis engine. Sound is acquired and synthesized by a similarity match between audio grains. A set of audio features are extracted offline for sound corpora and in real-time for live input.
The processing flow includes a non real-time part, where audio files are segmented, analyzed and ordered in a kD-tree structure, each grain being paired to a list of k-nearest neighbors. The system then performs a real-time feature extraction on live input, in order to find the best match in the tree. A pool of k grains is defined for the synthesis, exploiting the k-NN list coupled to this best match.

Keywords

Concatenative sound synthesis, real-time audio mosaicing, feature extraction.

1 Introduction

Sound synthesis techniques can produce countless timbres. Some of these create novel synthetic sounds; others aim to reconstruct defined spectra, often through a pre-analysis on existing sounds. Granular synthesis appears among the first applications in the field of sound production and reconstruction [1]. Unlike other sound synthesis techniques now crystallized in time, granular synthesis is still evolving in many musical applications, opening new perspectives and issues, concatenative sound synthesis (CSS) and audio mosaicing being an excellent example. The core idea of sound similarity synthesis is to extract descriptors from an audio file, to construct a timbral space and to introduce a metric such that the concept of distance becomes meaningful. The feature extraction highlights the perceptual similarities between sounds in a geometrical fashion. In this case, adjacent grains share similar sonic qualities in the morphological space. This space is multidimensional and its dimensionality depends on the number and the type of the descriptors used in the analysis. A non real-time feature extraction is performed for building a multidimensional database. In real-time, the incoming sound is analyzed using the same set of audio descriptors used in the offline analysis and the nearest neighbor search is carried out on this database. Lastly, a granular synthesizer plays back samples from a pool of sound units, constituted by the k-nearest neighbors (k-NN) of the best match. This part of the process is commonly referred to as synthesis.

The main motivation of this work is to develop a concatenative analysis-synthesis engine. The idea is to develop this system in Pure Data [1] environment giving the composer or the computer music designer the possibility of easily integrating this system into his programs dedicated to the processing of sound in real-time. The purpose of the project is also to create an open-source tool to perform contemporary music as in other commercial musical computing frameworks.
The main contribution of this work is to provide all the functionalities needed for the CSS in a single object, without the use of different tools for the analysis and the synthesis. This provides a more compact system that manages internally all the distinct steps. In addition, the internal algorithm is designed for the real-time application. Moreover other operations are performed offline, such as the database sorting and the calculation of k-nearest neighbors, that is needed in the synthesis part of the algorithm. Search on large database, i.e. long audio files, can be executed in a reasonable time.
The paper is organized as follows: the next section

[1] I. Xenakis' *Concret PH* is one of the first musical example.

describes the related works, focusing on projects developed in Max/MSP and Pd frameworks. Section 3 describes the methodology; in particular we focus on the proposed algorithm and on the specific implementation with some reference to technical decisions taken about the code. Results, both technical and artistic, are presented in Section 4. Finally we discuss the implementation and the future work.

2 Related Work

After the pioneering work of Iannis Xenakis, the attention of composers and researchers on granular synthesis and granulation of sampled sounds reached a point that, nowadays, granular synthesis is one of the most used techniques in electronic music composition.
However, several projects were developed using techniques involving concatenative synthesis from 2000s. In particular, corpus-based concatenative synthesis was explored with CataRT [3].

CataRT system is a set of patches for Max/MSP using several extensions, like FTM, Gabor and MnM. The idea is to load sound descriptors from SDIF [2] files, or to calculate descriptors on-the-fly, and to create a descriptors space using short grains as points. The user can choose two descriptors and using implemented displays, like graphic canvas, CataRT plots a 2-dimensional projection of grains in this descriptors space, where the user can navigate and explore the descriptors space through the mouse position. Within this graphical environment, the expressive nature of the grains can be emphasized. The CataRT architecture can be divided in five parts, as follows: analysis, data, selection, synthesis and interface. Apart from the interface facilities that we have previously discussed, this system uses other integrated Max/MSP tools like Gabor library for synthesis; in addition FTM data structures were used for data management. A piece for banjo and electronics by Sam Britton was composed within CataRT.

Another Max/MSP project was a piece of Marco Antonio Suárez Cifuentes: Caméléon Kaléidoscope [4], premiered at the Biennale Musiques en Scéne in Lyon in march 2010. A system combining FTM, Gabor, MnM and MuBu libraries was implemented for this production. MuBu is a multi-track container representing multiple temporally aligned homogeneous data streams similar to data streams represented by the SDIF standard [5].

[2]Sound Description Interchange Format.

In Pd world, several works were proposed on timbre analysis and sound classification. One of the largest projects is timbreID [2] library developed by William Brent. This library is a collection of externals for batch and real-time feature extraction. In particular, together with the presence of a set of low level features, i.e. a single-valued descriptors, some high level features, i.e. a multiple-valued descriptors, were implemented, such as mel-frequency cepstrum and bark frequency cepstrum. The Author shows that this set of high level features has an higher percentage of matching compared to the low level ones for audio event detection and classification [6]. Furthermore, the Author discusses the use of multi-frame analysis compared to single-frame analysis and the use of combined descriptors. In particular, high level ones plus low level features increase the percentage of matching at fixed analysis time.

This peculiarity is of great interest in the sound similarity reconstruction, because of the possibility of a best recognition and therefore a matching improvement between the incoming sound and the database elements. In addition the collection of single-descriptor external, a general object called timbreID is introduced, allowing for the management of the information database.
Finally, a recent audiovisual work, implementing a gestural controlled interface, has been presented [7]. This project is based on Pd, the computer graphics library GEM and timbreID library.

3 Methods

The *path~* analysis-synthesis architecture is presented in Figure 1.

Ahead-of-Time Analysis *path~* first performs an offline analysis. After loading samples, either from mono audio files in hard disk or from Pd's array, *path~* segments the audio corpora in slice, creating the available grains. Two types of segmentations are possible; the first one is frame-based, i.e. given an arbitrary frame, *path~* cuts out the samples at given frame and performs the analysis; all the grains have the same length.

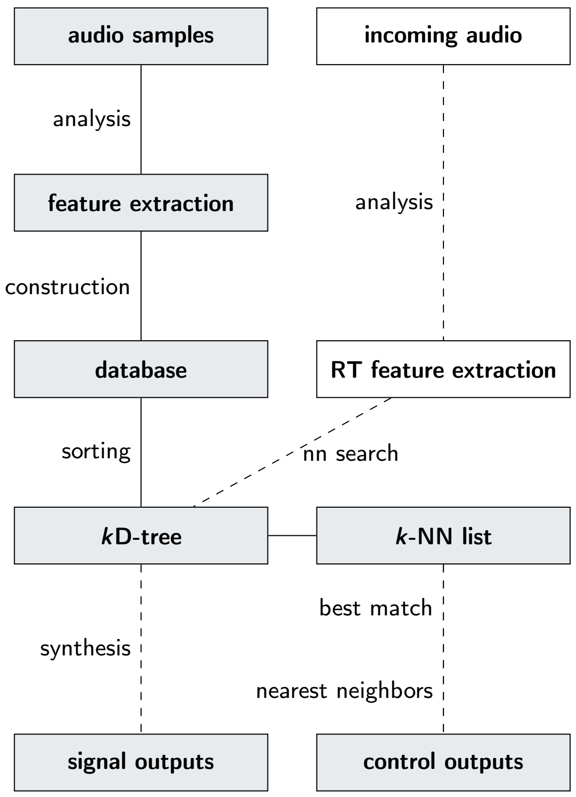

Figure 1: *path~* structure: ovals and dashed lines refer to real-time computations, rectangles and solid lines to offline analysis.

The second possible fragmentation is onset-based, i.e. an onset detection is made on the audio corpora and this fragmentation considers the gestural nature of audio files content. In this case, the grains have different lengths. Then features are extracted from all grains; these data form a database. We chose two multi-descriptors strategies. The default one is made by a high level descriptor plus a set of two low level features, while the second one excludes the high level descriptor leaving the others. The high level features is a mel-frequency cepstrum descriptor; while the low level features are a spectral centroid and an amplitude descriptor. The computation of Mel-Frequency Cepstral Coefficients requires a bank of overlapping triangular bandpass filters evenly spaced on the mel scale. A mel is the scale unit defined as:

$$Mel(f) = 2595 \times log_{10}(1 + \frac{f}{700}), \qquad (1)$$

where f is the frequency in Hz. Finally, in order to get the MFCCs we discrete cosine transform:

$$MFCC_i = \sum_{k=0}^{N-1} X_k cos[i(k + \frac{1}{2})\frac{\pi}{N}], \qquad (2)$$

with $i = 0, 1, \ldots, M-1$. M is the number of cepstral coefficients, N the number of filters and X_k is the log power output of the k^{th} filter. The user can choose the frequency of the mel spacing during the instantiation time. Varying this parameter, the number of resultant filters changes. This means that the dimension of our timbre space will change with respect to this parameter. The default value of mel spacing is 250 Hz given a 14-D timbre space [3], that is the number of MFCCs, plus 2, that is the dimensionality of the spectral centroid and of the amplitude descriptor: every grain will be represented as a set of 16 numbers, i.e the coordinates of a grain in the timbral space.
The spectral centroid (SC) is defined as the center of mass of magnitude spectrum.

$$SC = \frac{\sum_{k=0}^{N/2} f(k)|X(k)|}{\sum_{k=0}^{N/2} |X(k)|} \qquad (3)$$

with $f(k)$ the frequency of the k^{th} bin.
The amplitude descriptor is a root mean square amplitude by default; it can be changed in a peak amplitude.
path~ supports a multithreading strategy. In this way, the worker thread performs the offline analysis while the Pd thread continues its tasks. In this case, an offline analysis can be performed in an asynchronous manner without breaking the audio flow. However, our strategy partially breaks the Pd determinism. This is due to due to the fact the inter-thread communication is asynchronous and the analysis method is blocking only the worker thread function. The first attempt to alleviate this problem is to split the analysis method in two methods. The first one carries out the analysis while the second one performs the swap of the old data structure containing the needed data for the synthesis with the new one. The swap method lies in the Pd thread and it is synchronous with Pd. Moreover, we are dealing with this problem creating a format where user can store the data structure needed for the synthesis. This operation enables to simply load analysis that can be done in batch. This procedure reduces the time to get analysis complete. Finally, *path~* signals a bang message when the analysis is complete.
These are partially solutions to the problem and

[3]Tested mel spacings: 500 Hz = 6-D, 250 Hz = 14-D, 100 Hz = 38-D.

suggests as think to analysis as an ahead-of-time analysis. It can introduce several novel sound representations at the cost of relaxing Pd determinism.

Database The analysis forms a database, that is subsequently sorted in an indexed *k*D-tree. A *k*D-tree is a space-partitioning data structure for organizing points in a k-dimensional space. This data structure reduces the time complexity of nearest neighbor searches at the cost of tree construction. This sorting allows us to avoid linear searches giving the possibility of use large database. However, in high-dimensional spaces, the curse of dimensionality decreases the efficiency of the algorithm than in lower-dimensional spaces. The algorithm is only slightly better than a linear search of all of the points if the number of points is only slightly higher than the space dimensionality. For high-dimensional spaces, approximate nearest neighbor searches could be implemented.

Finally, *path~* constructs a *k*-NN list, that consists of available grains for the synthesis. This feature gives the user the possibility to have a pool of grains sharing sonic similarities that can be executed. Moreover, this list allows to have a fast and sequential access to the pool of grains without extra cost in real-time. The user can specify the maximum number of nearest neighbors to include in the pool. Due to the finite number of nearest neighbors usually requested, *path~* uses a quickselsort algorithm. Quickselsort is a partial sorting algorithm; it performs first a quickselect, a selection algorithm, then a quicksort, a sorting algorithm. This strategy allows us to decrease the cost of the *k*-NN list construction, that is executed for every grains.

Selection In real-time, *path~* performs a feature extraction on the live input. *path~* has the same analysis methods as for the source audio materials. However, there is no onset detection on live input; so even if the grains have different lengths the input analysis has a fixed length. The analysis is done according to the given user input and the windows analysis is independent on the blocksize. Then, *path~* searches in the *k*D-tree the element that minimizes the Euclidean distance. The selection is local, i.e. the best match is found individually. Accessing to the *k*-NN list, a pool of grains is available for the synthesis and *path~* outputs the control information. The control information is the matched index founded in the database, the list with the desired *k*-NN indices, the total number of grains and the number of grains actually played back.

Synthesis *path~* introduces a concatenation quality function. The term concatenation refers here to train length of the grain, i.e. the number of grains that are concatenated after the first one. In this way, a grains train can start creating sound textures. Accessing to the *k*-NN list, a certain amount of grains can be executed and concatenated. The user can access the list linearly or randomly obtaining different sonic results.

Preset Finally we implemented a script interpreter as preset system. In this way, the user can write a script within a given syntax and *path~*loads it and creates a preset that can be called with a dedicated method. Optionally, the user can specify an identificative name for every preset. Apart from the interpreter, a simple debugger has been added. Furthermore, the complete preset list containing the debugged list with the various preset names can be viewed in a GUI editor that can be called clicking on *path~*.

```
# path~ preset file example ;
# this is a comment ;

preset @ intro ;
threshold = 0.;
window = 512;
concatenate = 1;
amp = 1;
hopsize = 100;
weight = 1;
envelope = 0;
endpre

preset @ sec1 ;
window = 2048;
threshold = 0.25;
concatenate = 3;
amp = 0.2 0.4;
hopsize = 100 3000;
envelope = 2;
endpre
```

Preset file example.

4 Results

4.1 Latency

One of the main problems in real-time audio is the latency, defined as a time delay between the input arrival and the output response. So we study the time spent by *path~* between the user input and the grains response. Our estimate about real-time latency limit is 6 ms. On Authors' machine, i.e.

a Mac BookPro Early 2011 13" 2.7 GHz with Pd Vanilla 0.47.1, on a database size of the order of 30K obtained using a 10 minutes audio file long with a 23 ms analysis window, database lookups took less than 2 ms on average, while offline analysis took less than one minute on average.

4.2 Application in mixed contemporary composition

The use of real-time analysis systems for instruments' sound is a key field in the interaction between acoustic instruments and electronics. While over the last few years various methods of analysis of the sound signal, e.g. audio features extraction, have been implemented, these methods have not been explored in all them possibilities.
In the following, we discuss about three projects where *path~* is involved. In particular, we present the different strategies within which *path~* has been designed to fulfill the requests of composers and computer music designers.

On the fly

The first musical application was Marco Matteo Markidis' radical improvisation duo Adiabatic Invariants. The entire electronics of Cattedrali di Sabbia, a 40 minutes improvisation, is based on *path~*. One of the main problems was to use *path~* on the fly, due to the improvised nature of the duo. The multithreading idea was born in this way; moreover during the set several fragments of percussions are recorded on Pd's arrays and analyzed in the following. Moreover, in chaotic musical situation, an high number of small events can be very useful; we experimented several configurations with a lot of concatenated grains. Finally, *path~* was used for audio mosaicing. Audio mosaicing refers to the process of reconstructing the temporal evolution of a target sound from fragments taken from a source audio materials. In this case, *path~* was triggered with a metronome and in an asynchronous way by the use of an envelope follower.

Space

For *S4EF*, a work for augmented alto saxophone [4] and electronics by Giuseppe Silvi, *path~* is called in three instances at the same time: one for *Windback*, the electroacoustic device applied on saxophone; two for *S.T.ONE* tetrahedral loudspeakers.
The *Windback* augmentation consists in a feedback system between embouchure and bell of saxophone, with hardware and software control of both [9]. The Windback call of *path~* is monophonic with frame-based fragmentation to control fine morphology sounds for internal feedback. The two *S.T.ONE* calls of *path~* are polyphonic, each of four channels, one for each side of tetrahedral loudspeaker. The S.T.ONE system (*Spherical Tetrahedral ONE*) is a loudspeaker designed by Giuseppe Silvi, based on *Ambisonic* technology with the ambition to obtain electroacoustic diffusion of electronic music with the same space relationship and sound shape of acoustic instruments. The use of grain polyphony by this kind of technologies has been improved synthesizing contemporary grains on different loudspeakers. For S.T.ONE sound synthesis *path~* is used with onset detected fragments to obtain grains with emphasized musical content.
The Windback research team now is focused on full augmentation of saxophone quartet (SATB) where a multi-input version of *path~* will be an useful tool of sound synthesis for *D.R.Y.* augmented saxophone quartet by Giuseppe Silvi.

Live input

Currently, we are working with Lara Morciano, an Ircam composer, on her new piece for piano and electronics. One of the most interesting Morciano's requests is to use no sound as live input, but to investigate the timbre space directly with the use of motion caption. Our strategy is to access directly to the timbre space; it allows us to explore several configurations even with or without live sound. Technically, the incoming motion captured data are added to the features extracted ones; basically it is possible to change the coordinates of the live extracted point to be used in the database lookup. With no sound, these data act as an offset respect to space origin. This gestural controlled way to produce grains is particularly effective with onset detected grains in such a way that gestures are related to musical meaningful grains. For this piece, we are using *path~* without the high level descriptors; the captors communicate directly with the two low level descriptors. This strategy allows for a more clear control.

[4]The Windback system is made by Michelangelo Lupone; the *S4EF* work is produced by *CRM Centro Ricerche Musicali*, Rome.

5 Discussion and Conclusions

The primary aim of this work is the production of contemporary music pieces for acoustic instruments and electronics. To achieve this aim, several improvements are planned. In particular a multi-input version will be useful in an ensemble context. In addition, a greater control on spatialization and on single grain control localization can be desirable. Moreover, an editor improvement together with a more detailed preset configuration and setup will be designed. A growing interest will be dedicated to cluster analysis implementation and on pattern recognition, particularly trying to develop some specific strategy for extended contemporary techniques and for different grain lengths. Finally, a forthcoming research is a multi-agent approach to corpus-based CSS [8] allowing us to explore the morphological space.
path~ has been tested on OS X and Linux using Vanilla stable release 0.47.1. *path~* is an open source software and it is released under GPLv3. It is available via Deken, the externals wrangler for Pure Data.

6 Acknowledgements

We would like to thank Stefano Markidis for his suggestions; Michele Orrù and Ilaria Ciriello for the technical guidance and encouragement; Matt Barber and Jonathan Wilkes for the stimulating discussions; Miller Puckette, William Brent and the cyclone library team for the knowledge their codes gave us. We thank entire Pd community: great community, great software, great freedom.

References

[1] M. Puckette: *Pure Data: another integrated computer music environment*, Proceedings of the International Computer Music Conference (ICMC), 1996.

[2] W. Brent: *A timbre analysis and classification toolkit for Pure Data*, Proceedings of the International Computer Music Conference (ICMC), 2009.

[3] D. Schwarz and G. Beller and B. Verbrugghe and S. Britton: *Real-time corpus-based concatenative synthesis with CataRT*, Proceedings of the International Conference on Digital Audio Effects (DAFx), 2006.

[4] N. Schnell and Marco Antonio Suàrez Cifuentes and Jean-Philippe Lambert: *First steps in relaxed real-time typo-morphological audio analysis/synthesis*, Proceedings of the Sound and Music Computing Conference (SMC), 2010.

[5] N. Schnell and A. Röbel and Diemo Schwarz and Geoffroy Peeters and Riccardo Borghesi: *Mubu & Friends - Assembling tools for content based real-time interactive audio processing in Max/MSP*, Proceedings of an International Computer Music Conference (ICMC), 2009.

[6] W. Brent: *Cepstral analysis tools for percussive timbre identification*, Proceedings of International Pd Conference (PdCon09), 2009.

[7] J. Gossmann and M. Neupert: *Musical Interface to Audiovisual Corpora of Arbitrary Instruments*, Proceedings of New Interfaces for Musical Expression (NIME), 2014.

[8] D. Pozzi: *Composing exploration: a multi-agent approach to corpus-based concatenative synthesis*, Proceedings of Colloquium on Musical Informatics (CIM), 2016.

[9] M. Lupone, S. Lanzalone, L. Seno: *New advancements of the research on the augmented wind instruments: Wind-back and ResoFlute*, Electroacoustic Winds Conference, Aveiro. 2015.