A Two-Dimensional Live Playback Sampler-Looper based on Loudness and Spectral Shape descriptors

Marco Fiorini
mfiori21@student.aau.dk
Aalborg University Copenhagen
Sound and Music Computing, 8th Semester
Sound and Music Signal Analysis

***Abstract*—In this project, a two-dimensional corpus exploration patch in Max/MSP have been implemented, based on different sound and music signal analysis techniques. A sound file is segmented using Onset Detection and analysed with Loudness and Spectral Shape descriptors from the FluCoMa library. This information is then used to drive a looping playback mechanism, similar to an automated sequencer. The notion of descriptor-driven playback is then expanded: the descriptor values are processed and made usable as coordinates so that each sound segment can be placed on to a plotter canvas. This creates the visual space that can be navigated with the mouse as a live playback sampler-looper.**

I. Introduction

In the last years, new creative ways of using sound descriptors in live electronic performances have been developed, expecially in the Max/MSP environment [1]. Significant contributions to this field come from the STMS (Science and Technology of Music and Sound) research lab at IRCAM in Paris [2], where complex toolkits and applications have been developed, like MuBu [3] or Somax2 [4]. Building upon these applications, a notable recent contribution, particularly in the field of improvisation, have been given by Jérôme Nika and Gérard Assayag [5] in the context of the DYCI2 project [6], introducting generative agents and tools for human-machine live interaction to compose at the behavioural level and activate the performance by interaction. The interesting innovation highlighted by this research consists on the definition of a *corpus*, a set of data extrapolated by audio descriptors and triggered by memory recollection.

Trying to create a connection between the curriculum of the Sound and Music Signal Analysis course and my personal trajectory, I got very interested in the FluCoMa project. The Fluid Corpus Manipulation project (FluCoMa) "instigates new musical ways of exploiting ever-growing banks of sound and gestures within the digital composition process, by bringing breakthroughs of signal decomposition DSP and machine learning to the toolset of techno-fluent computer composers, creative coders and digital artists" [7]. From this community, a wide number of inspiring applications using sound descriptors for assisted composition or improvisation have been published, like the works of Lauren Sarah Hayes on event detection in improvisation [8] or Alex Harker's on exploring the multiphonic possibilities of the oboe [9].

Thus, as an improvising musician in the area of electroacoustic and electronic music, this project stands out as a personal starting point in the research of sound descriptors for creative applications in assisted composition and corpus-based improvisation of live electronic performances.

II. Problem Analysis

A. Sound Analysis with Audio Descriptors

A wide number of studies have been performed in the field of sound classification, both in terms of speech and instruments [10], using sound descriptors, resulting also in many psico-acoustic studies [11]. Sound or audio descriptors (or features) are contextual pieces of information about the contents of an audio signal. Descriptors may be either symbolic or numerical (discrete or continuous) and can be extracted directly from the audio signal. The purpose of an effective descriptor is to encode meaningful information about the audio signal for use in applications where the algorithm needs to be content-aware. As described by Peeters in [12], audio descriptors can be distinguished according to a systematic taxonomy, based on:

- steadiness or dynamicity,
- time extent of the description provided,
- abstraction,
- extraction process.

While a detailed investigation of the taxonomy is beyond the scope of this research, I believe that a further highlight of the time extend validity may be useful. In the context of how the features are applied to a signal over time, we can distinguish between two variables:

- *Global descriptors*: computed for the whole signal (e.g. the attack duration of a signal). These descriptors require to have a previous time localization of the sound events.
- *Instantaneous descriptors*: computed for each time frame (a time frame is a short time segment of the signal, around

60 msec of length). Example of this are the spectral centroid of a signal which varies along time, as used in this research.

Furthermore, according to the different extraction process, a wide set of descriptors could be schematized, with features related to:

- *Temporal shape*,
- *Temporal features*,
- *Energy features*,
- *Spectral shape features*,
- *Harmonic features*,
- *Perceptual features*.

In this project, I decided to focus on two distinct instantaneous features, extracted after a procedure of Onset detection, namely Loudness and Spectral Shape descriptors. The following sections will provide an introduction and definition of these concepts.

B. Onset Detection

Onset detection is a well-defined task, aiming to find the starting time of each musical note (where a musical note is not restricted to those having a clear pitch or harmonic partials) [13]. In a recent tutorial article, Bello et al. [14] reviewed a number of onset detection algorithms using spectral features such a magnitude, phase and complex domain representations, making a theoretical and empirical comparison of several state-of-the-art approaches. Following this work, Dixon [15] proposed improvements to these methods, like the Rectified Complex Phase Deviation used in this project implementation. Nevertheless, in order to talk about this method, a couple of previous onset detection functions (a function whose peaks are intended to coincide with the times of note onsets) need to be introduced. As proposed in [15], all of these methods make use of a time-frequency representation of the signal based on a Short Time Fourier Transform (STFT) using a Hamming window $w(n)$. If $X(k,l)$ represents the kth frequency index (or bin) of the lth frame index, then:

$$X(k,l) = \sum_{n=-N/2}^{N/2-1} w(n)x(n+lL)e^{-\frac{2j\pi nk}{N}} \tag{1}$$

where N is the window length and L the hop size [16].

1) Spectral Flux: Spectral flux measures the change in magnitude in each frequency bin, and if this is restricted to the positive changes and summed across all frequency bins, it gives the onset function SF [15] [17]:

$$SF(l) = \sum_{k=-N/2}^{N/2-1} H(|X(k,l)| - |X(k,l-1)|) \tag{2}$$

where $H(x) = \frac{x+|x|}{2}$ is the half-wave rectifier function.

2) Complex Domain: Amplitude and phase can be considered jointly to search for departures from steady-state behaviour by calculating the expected amplitude and phase of the current bin $X(k,l)$, based on the previous two bins $X(k,l-1)$ and $X(k,l-2)$. The target value $X_T(k,l)$ is estimated by assuming constant amplitude and rate of phase change:

$$X_T(k,l) = |X(k,l-1)|e^{\phi(k,l-1)+\phi'(k,l-1)} \tag{3}$$

and therefore a complex domain onset detection function CD can be defined as the sum of absolute deviations from the target values [15]:

$$CD(l) = \sum_{k=-N/2}^{N/2-1} |X(k,l)| - |X_T(k,l)| \tag{4}$$

3) Rectified Complex Phase Deviation: As reported by Dixon in [15], one problem with the CD method is that it does not distinguish between increases and decreases in amplitude of the signal. Since it is important to distinguish onsets from offsets, Dixon [15] proposed then a similar idea to that used in the SF function, where half-wave rectification is used to preserve only the increases in energy in spectral bins. This idea can easily be incorporated into the CD method, giving a (half-wave) rectified complex domain (RCD) onset detection function as follows:

$$RCD(l) = \sum_{k=-N/2}^{N/2-1} RCD(k,l) \tag{5}$$

where

$$RCD(k,l) = \begin{cases} |X(k,l)| - |X_T(k,l)|, & \text{if } |X(k,l)| \geq \\ & |X(k,l-1)| \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

C. Loudness Features

Our perception of how loud a sound is can be complex to understand and measure. Human hearing is not constant or linear across the range of frequencies that we are sensitive to. It is possible to have sounds that register as loud when measured that nonetheless are perceived as weak (and vice versa) depending on their frequency content. Loudness is an audio descriptor that attempts to model such characteristics of human hearing according to the EBU R128 specification [18]. This specification uses an approach that both respects the natural filtering of our ears across the audible spectrum of frequencies and incorporates a notion of differing time scales into the analysis.

From an analytical standpoint, Bark bands [19] can model a better approximation of the Human Auditory System and are used for the calculation of Perceptual descriptors like Loudness, Specific Loudness, Sharpness and Spread. The Bark scale ranges from 1 to 24 Barks, corresponding to the first 24 critical bands of hearing [20], as shown in Figure 1.

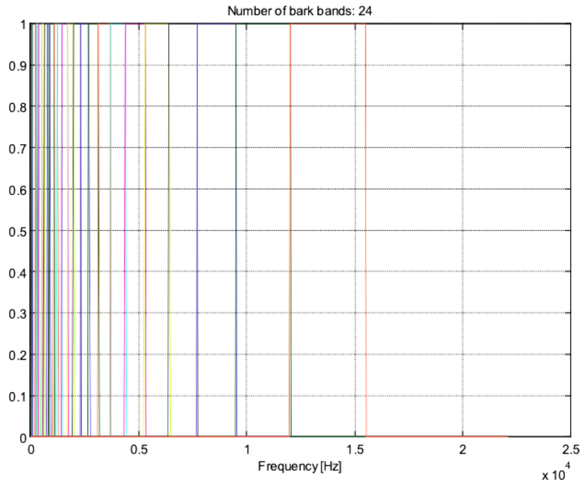


Fig. 1. Bark bands of frequencies.

The specific loudness is the loudness associated to each Bark band. The precise expression of the loudness can be found in [21]. In [12] Peeters gave an approximation of it, by neglecting the terms of the expression acting only in specific cases, like very weak signals. Noting as $N'(z)$ the loudness in the zth Bark band, the *specific loudness* can be expressed in a relative scale as:

$$N'(z) = E(z)^{0.23} \tag{7}$$

The *total loudness* could then be computed as the sum of individual loudnesses:

$$N = \sum_{1}^{bands} N'(z) \tag{8}$$

Additionally, *relative specific loudness* could be defined as the specific loudness normalized by the total loudness as:

$$Nrel(z) = N'(z)/N \tag{9}$$

This measurement could be useful in terms of LUFS (Loudness Units relative to Full Scale) calculation, as defined in EBU R128 [18] and widely used as the industry standard in broadcasting and mastering of commercial audio productions, representing the average loudness level of a signal over a given time period based on human perception of loudness.

D. Spectral Shape Features

Spectral Shape are usually instantaneous features computed from the Short Time Fourier Transform (STFT) of the signal. A spectrogram is a visual representation of the spectrum of frequencies of a signal as it varies with time (as shown in Figure 2) and it is computed as the magnitude spectrum of the STFT (as previously defined in Equation 1):

$$S_x(k,l) = |X(k,l)|^2 \tag{10}$$

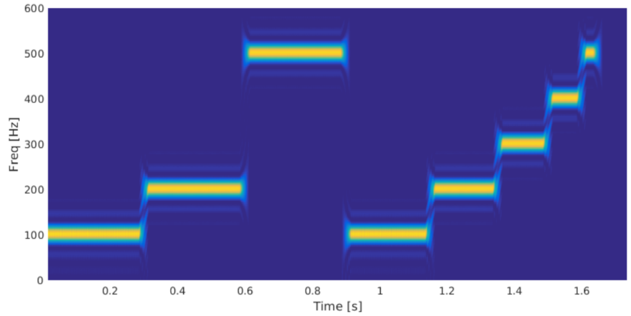


Fig. 2. Spectrogram of a time-varying sinusoid, with frequencies displayed over time.

Examples of these descriptors are Spectral Centroid, Spread, Skrewness, Kurtosis, Slope, Roll-on frequency and Mel-Frequency Cepstral Coefficients (MFCC). Despite many of these descriptors are available in the implementation opportunities of this project, the final implementation relied on Spectral Centroid. Therefore, a description of this specific classifier will be given.

1) Spectral Centroid: The Spectral Centroid is the barycenter of the spectrum. It is computed considering the spectrum as a distribution which values are the frequencies and the probabilities to observe these are the normalized amplitude.

$$\mu = \int x \cdot p(x) \delta x \tag{11}$$

where x are the observed data and $p(x)$ the probability to observe them.

III. Implementation

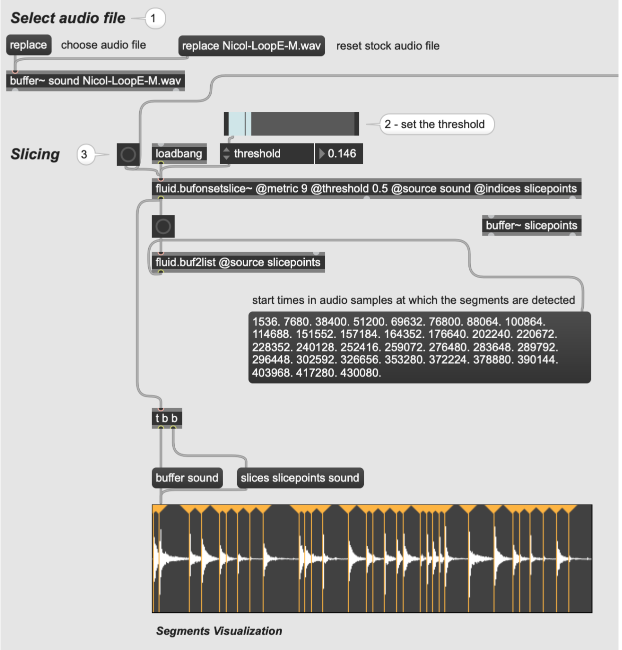
For the implementation of this project I used Max/MSP [1] and the FluCoMa Fluid Manipulation Toolkit, a software consisting of objects for decomposing and describing audio, and for manipulating collections of sonic data by querying, matching, learning and transforming [7].

A. Onset Slicing

To perform onset detection and slicing, the object `fluid.bufonsetslice∼` has been used. OnsetSlice calculates slice points in a sound based on changes in its spectrum. It makes these judgements based on one of several available metrics, which each have their own characteristics. From this, it derives a time-series that describes change from one moment to the next. If that change exceeds the given threshold parameter, a slice point for that event is produced.

There are 10 different metrics that OnsetSlice can use for calculating the changes in the spectrum [22]:

- Metric 0 (Energy): Difference between sum of squares of the magnitude of frames. Good for material that has large differences in amplitude.
- Metric 1 (High Frequency Content): Similar to metric 0 except that each frequency bin is weighted by the bin number. This means that the high frequency content is weighted more strongly in the calculation of difference.
- Metric 2 (Spectral Flux): Measures difference between the magnitudes of spectral frames. A popular algorithm and a suitable choice for a variety of sound types.
- Metric 3 (Modified Kullback-Leibler): Similarly to metrics 6, 7, 8, 9, metric 3 compares a projection of the spectrum based on frames in the past and compares the reality to this projection. Importantly, this uses log magnitudes per bin, whereas most other metrics don't.
- Metric 4 (Itakura-Saito): Uses the Itakura-Saito divergence to measure the difference between the past spectrum and an approximate projection from that into the future [23]. In Non-negative matrix factorization (NMF), the Itakura-Saito divergence can be used as a measure of the quality of the factorization: this implies a meaningful statistical model of the components and can be solved through an iterative method [24].
- Metric 5 (Cosine): Calculates the difference between spectral frames using cosine distance.
- Metric 6 (Phase Deviation): Phase deviation uses past spectral frames to projects what the next frame from those will be. It then compares the projection to the reality. If the difference between projection and reality exceeds the threshold, a slice is output. This is especially useful for sounds which alternate or change between stable and unstable sonic states. An example of this might be sinusoidal or tonal material that rapidly becomes noisy.
- Metric 7 (Weighted Phase Deviation): Very similar to metric 6, except the phase is weighted by the magnitude.
- Metric 8 (Complex Phase Deviation): The same as metric 6 but calculated in the complex domain.
- Metric 9 (Rectified Complex Phase Deviation): The same as metric 8 but rectified.



```
buffer~ sound Nicol-LoopE-M.wav
fluid.bufonsetslice~ @metric 9 @threshold 0.5 @source sound @indices slicepoints
buffer~ slicepoints
fluid.buf2list @source slicepoints
t b b
buffer sound
slices slicepoints sound
```

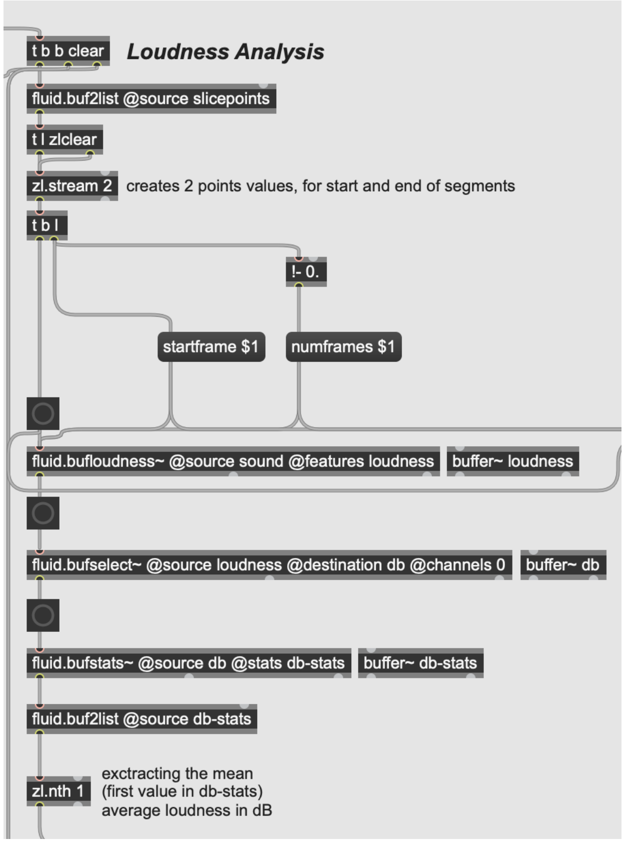
```
1536. 7680. 38400. 51200. 69632. 76800. 88064. 100864.
114688. 151552. 157184. 164352. 176640. 202240. 220672.
228352. 240128. 252416. 259072. 276480. 283648. 289792.
296448. 302592. 326656. 353280. 372224. 378880. 390144.
403968. 417280. 430080.
```

Fig. 3. First section of the Max/MSP patch, showing the process of slicing using Onset Detection. The audio file is loaded in a buffer and fed to the fluid.bufonsetslice object with @metric argument 9 (RCD). The resulting slicepoints are visualized as segments over the original audio file.

In this application, Metric 9 (Rectified Complex Phase Deviation) has been used. This because, as reported in [15], this method works well with a wide range of sounds and its implementation has a normalised threshold range, with values between 0.0 and 1.0.

B. Loudness Analysis

Like most descriptors, the `fluid.bufloudness∼` object works over a short configurable window of time. For instant loudness, the EBU standard recommends a window size of 400ms that is updated every 100ms [18] [25]. For each window, the object provides a measurement of loudness of the signal (in LUFS) and also provides the true-peak of that window (in dBFS). With these pieces of information, we might hope that our musical systems and programs can "listen" to sounds and infer their loudness in a way that is more similar to how we might perceive them. This is a murky area to untangle when a computer is involved, and our biological processes responsible for hearing are subject to all sorts of other influences and contextual pieces of information. Aspects such as fatigue and expectation play a large role in the perceived loudness of a sound. In other words, it is important to appreciate the model that is being used, while being aware that it can fall short for replicating complex human hearing.



```
t b b clear
fluid.buf2list @source slicepoints
t l zlclear
zl.stream 2
t b l
!- 0.
startframe $1
numframes $1
fluid.bufloudness~ @source sound @features loudness
buffer~ loudness
fluid.bufselect~ @source loudness @destination db @channels 0
buffer~ db
fluid.bufstats~ @source db @stats db-stats
buffer~ db-stats
fluid.buf2list @source db-stats
zl.nth 1
```

Loudness Analysis
creates 2 points values, for start and end of segments
exctracting the mean
(first value in db-stats)
average loudness in dB

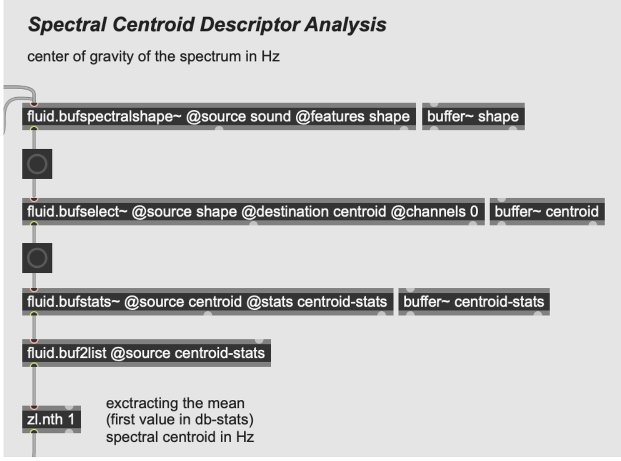
Fig. 4. Loudness Analysis using the FluCoMa library. The audio buffer is converted to a list, to create two-point values, for the beginning and the end of each segment previously sliced with the onset detection. These frames are then send to the `fluid.bufloudness∼` object, computing the true peak of the signal as well as applying the filters proposed by broadcasting standards to emulate the perception of amplitude.

The FluCoMa loudness analysis, as shown in Figure 4, is based on true peak, a measurement that takes into account inter-sample peaks by means of oversampling the signal and gives the result for a specific point in time. This is because most digital meters display sample peaks (the highest number identified in a given time window). But when a digital audio is played back, what is heard is actually an analog reconstruction of that sound as the digital data is converted into an analog signal for the loudspeakers. The reconstructed analog signal can, in certain conditions, peak beyond the maximum sampled values: these are known as inter-sample peaks.

C. *Spectral Analysis*

`fluid.bufspectralshape∼` is an object that calculates several audio descriptors and bundles them together. This collection of audio descriptors describe the shape of a spectrum, giving information about the characteristics of a sound. It can give indications about how spread out a spectrum is, where the centre is or perhaps how flat or tilted the overall shape is. As always, audio descriptors have the potential to be misleading, therefore, it is important to keep in mind how the values themselves are derived so that we can apply our musical judgement to them and derive usefulness from what they are actually describing about the spectrum.

- Spectral Centroid: Centroid is the centre of gravity or centre of mass in the spectrum.
- Spectral Spread: The spectral spread describes the amount of Deviation of the energy around the spectral centroid.
- Spectral Skewness: How skewed, or symmetrical the spectrum is around the mean.
- Spectral Kurtosis: How peaky or pointy the spectrum is.
- Spectral Rolloff: The frequency below which is contained 99% of the energy of the spectrum.
- Spectral Flatness: Literally how flat the spectrum is. White noise occupies equal energy in each frequency bin of a spectral analysis and is therefore very flat. A single sinusoidal peak will be not flat at all. It might be useful for differentiating noisy and tonal sounds.
- Spectral Crest: Crest is the ratio between the loudest magnitude and the RMS of the analysis frame. A larger number is an indication of a loud peak poking out from the overall spectral curve



```
fluid.bufspectralshape~ @source sound @features shape
buffer~ shape
fluid.bufselect~ @source shape @destination centroid @channels 0
buffer~ centroid
fluid.bufstats~ @source centroid @stats centroid-stats
buffer~ centroid-stats
fluid.buf2list @source centroid-stats
zl.nth 1
```

Spectral Centroid Descriptor Analysis
center of gravity of the spectrum in Hz
exctracting the mean
(first value in db-stats)
spectral centroid in Hz

Fig. 5. Spectral shape analysis using the FluCoMa library. First, the object `fluid.bufspectralshape∼` calculates all the seven spectral features described in Section III-C. Then, centroid is selected through `fluid.bufselect∼` and the resulting frequencies are displayed in increasing order to feed the plotting square.

In this implementation, Spectral Centroid has been chosen as an instantaneous descriptor computing the barycenter of the spectrum.

D. *Plotting*

At this point of the implementation, the descriptor values are processed and made usable as as coordinates so that each sound segment can be placed on to a `fluid.plotter` canvas, creating the visual space that can be navigated with the mouse.

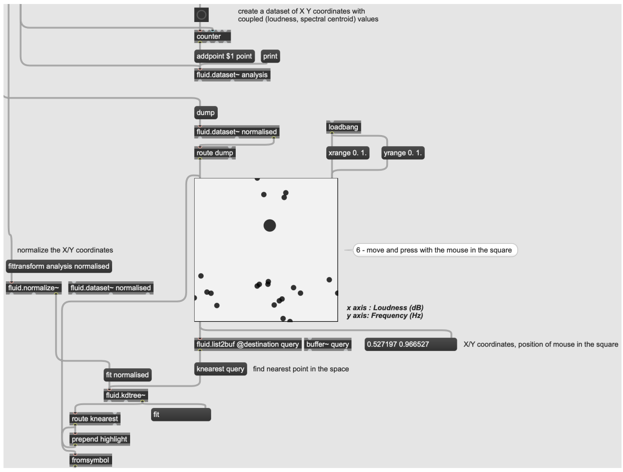

create a dataset of X Y coordinates with
coupled (loudness, spectral centroid) values
counter
addpoint $1 point
print
fluid.dataset~ analysis
dump
fluid.dataset~ normalised
loadbang
route dump
xrange 0. 1.
yrange 0. 1.
6 - move and press with the mouse in the square
normalize the X/Y coordinates
fittransform analysis normalised
fluid.normalize~
fluid.dataset~ normalised
x axis : Loudness (dB)
y axis: Frequency (Hz)
fluid.list2buf @destination query
buffer~ query
0.527197 0.966527
X/Y coordinates, position of mouse in the square
knearest query
find nearest point in the space
fit normalised
fluid.kdtree~
fit
route knearest
prepend highlight
fromsymbol

Fig. 6. Plotting of the sliced samples on a two-dimensional square using `fluid.plotter`. Here, the selected sample is highlighted, resulting bigger than the others. The samples are displayed in relation to their distance according to the analyzed descriptors (X axis: loudness, Y axis: spectral centroid).

As presented in Figure 6, the values obtained from loudness and spectral analysis are joined in a buffer where every point share the same starting and ending position of the sliced segment from the original audio file. This operation is crucial to create then a `fluid.dataset∼` of X-Y coordinates with coupled values of loudness and spectral centroid. The dataset is then normalized, resulting in a two-dimensional space of coordinates ranging from 0 to 1. Finally, the dataset is routed to the plotting space where the values are displayed, according to their distance in terms of the analyzed descriptors, with loudness (in dB) on the X axis and frequency (in Hz) on the Y axis.

IV. Results

To present the application in a user-friendly context, the patch has been featured with a presentation mode, where only the object essential for the interaction are displayed. As show in Figure 7, an audio file can first be selected. Then, the threshold value for the onset detection and slicing can be set. At this point, the samples are displayed in the plotting square where they can be selected (the highlighted one will always result slightly bigger than the others). As it is noted, the slicing button needs to be pressed twice to correctly display the segments in the plotting canvas. This is not due to the onset detection algorithm but rather to an internal routing of signal that first slices the audio buffer and then plots the sliced samples to the square. After activating the audio segments playback, the user can navigate the plotting canvas and select in real time different audio samples to be played in loop.

A video demo of the application in the context of two different audio files is available[1].

V. Discussion

The goal of this research was to investigate possible creative uses of sound analysis through audio descriptors and give a concrete real time application to enhance compositional and improvisational practices of electronic music performance. Specifically, a real time patch providing loudness and spectral shape analysis of audio samples previously sliced through an onset detection alogrithm has been developed in Max/MSP. The result is a software tool that gives immediate control of audio sampling, it is modular and open to future extensions and a wide set of applications.

Looking at the demo presented in the video, we can notice that longer files need a slightly longer period of time to be analyzed by the whole algorithm chain and thus for the samples to be displayed in the plotting square. Nevertheless, this waiting is in terms of seconds and it does not seem to affect the user's flow in a real time performance. The looping mechanism is well responsive to the changes of the selected samples, providing a very wide number of possible performance applications. Another important thing to notice is that, depending on the set threshold for the onset detection, the samples will have a different duration. This can be changed and normalized, if the user needs a uniform length of samples, by chunking the longer samples, but if short samples are needed, the best solution is just to select a lower level of threshold.

As previously stated in Section III-D, the usage of `fluid.plotter` is only one of the several possibilities of interacting with the obtained data. Further explorations of the same implementation could be investigated using the `fluid.grid∼` object. This maps a set of two-dimensional points to a rectangular grid of evenly spaced values. This implementation would turn to be really useful if the user wants to control the playback of the samples with an external controller, like Ableton Push [26] or Novation Launch [27]. The advantage of this interface coupling would consist in an intuitive real time performance based on a well known physical user interface with extended possibilities of sound analysis. In fact, the proposed implementation is open to modular augmentation, adding more features based on different descriptors, keeping the possibility of interfacing with an external device as just stated.

VI. Conclusions

The present implementation of sound analysis techniques contributes to a larger context of studies on audio descriptors and creative usage of them. It is hoped that the research done during this study can work as a personal foundation for future investigations in these fields, as well as in the area of tools for digital assisted composition and improvisation.

[1] https://drive.google.com/file/d/1IPsMGhM7W15XT_l6s9o8O2Ei1r_c_qeH/view?usp=sharing

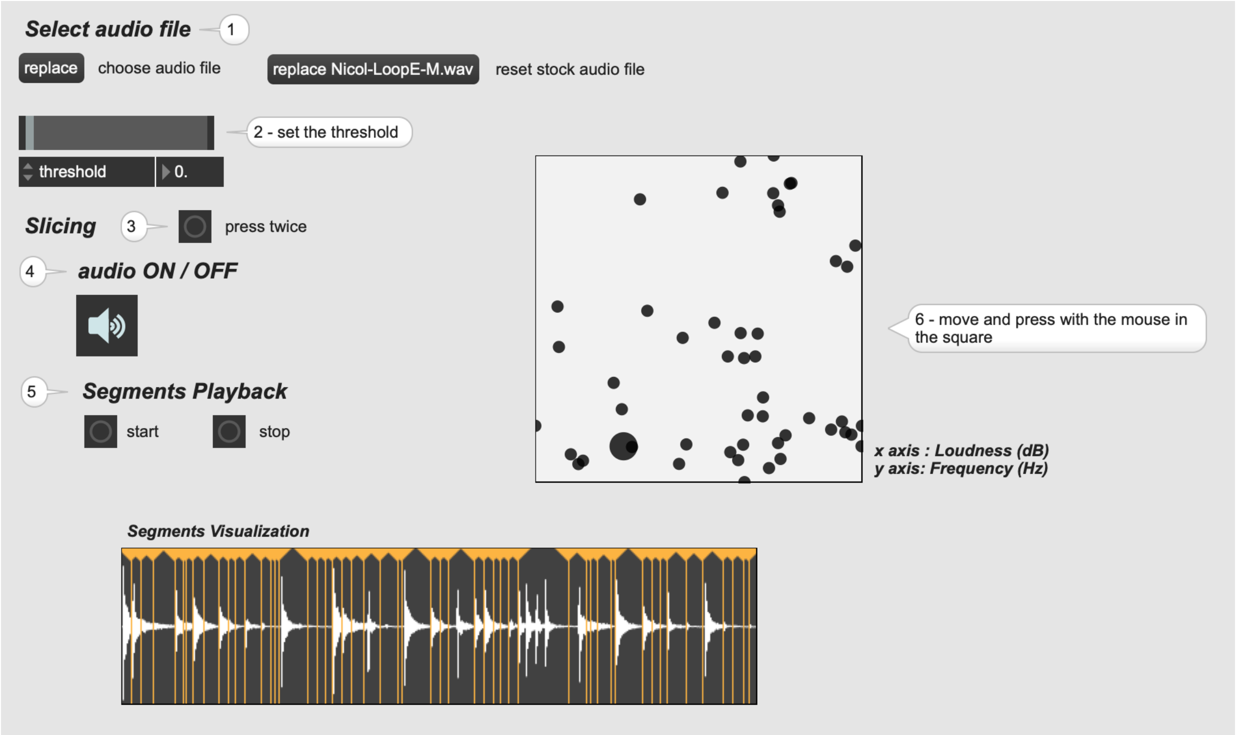


Fig. 7. The complete Max/MSP patch in presentation mode. This consists of an essential overview of the main elements of the project, to make the interaction more intuitive. The user can first select a desired audio file, or use the stock one, consisting of a drum recording. Then the threshold for the onset detection and slicing is set. After the slicing, the audio playback can begin, as the user can press and move the mouse inside the plotting square to select a different sample to be played in loop.

References

[1] "Max/MSP, Cycling '74." [Online]. Available: https://cycling74.com/

[2] "IRCAM." [Online]. Available: https://www.ircam.fr/

[3] "Mubu | ircam forum." [Online]. Available: https://forum.ircam.fr/projects/detail/mubu/

[4] "Somax2 | stms lab." [Online]. Available: https://www.stms-lab.fr/projects/pages/somax2/

[5] "Music generation: Composing behaviours – jérôme nika." [Online]. Available: https://jeromenika.com/research-designing-generative-agents/generation-impro/

[6] "Releases · dyci2/dyci2lib · github." [Online]. Available: https://github.com/DYCI2/Dyci2Lib/releases

[7] "FluCoMa." [Online]. Available: https://www.flucoma.org/

[8] L. S. Hayes, "Event detection and improvisation." [Online]. Available: https://learn.flucoma.org/explore/hayes/

[9] A. Harker, "Exploring the oboe with flucoma." [Online]. Available: https://learn.flucoma.org/explore/harker/

[10] J. C. Brown, "Musical Instrument Identification using Autocorrelation Coefficients," in *Proceedings International Symposium on Musical Acoustics ISMA 1998,Leavenworth, Washington*, June 1998, pp. 291–295.

[11] J. Krimphoff, S. McAdams, and S. Winsberg, "Caracterisation du timbre des sons complexes. ii. analyses acoustiques et quantification psychophysique," *Journal De Physique*, vol. 4, pp. 625–628, 5 1994.

[12] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the CUIDADO project," Icram, Tech. Rep., 2004.

[13] S. Hainsworth and M. Macleod, "Onset detection in musical audio signals."

[14] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler, "A tutorial on onset detection in music signals," *IEEE Transactions on Speech and Audio Processing*, vol. 13, pp. 1035–1046, 9 2005.

[15] S. Dixon, "Onset Detection Revisited," *Proc. of the 9th Int. Conference on Digital Audio Effects DAFx'06, Montreal, Canada*, 2006.

[16] M. M. Goodwin, "The stft, sinusoidal models, and speech modification," *Springer Handbooks*, pp. 229–258, 2008.

[17] P. Masri, "Computer modelling of sound for transformation and synthesis of musical signals," 1996. [Online]. Available: http://www.fen.bris.ac.uk/elec/dmr

[18] "Loudness | ebu technology innovation." [Online]. Available: https://tech.ebu.ch/loudness/

[19] E. Zwicker and I. E. Terhardt, "Analytical expressions for critical-band rate and critical bandwidth as a function of frequency," *The Journal of the Acoustical Society of America*, vol. 68, p. 1523, 8 1998. [Online]. Available: https://asa.scitation.org/doi/abs/10.1121/1.385079

[20] H. Fastl and E. Zwicker, "Psychoacoustics: Facts and models," *Psychoacoustics: Facts and Models*, pp. 1–463, 2007.

[21] b. c. j. moore, b. r. glasberg, and t. baer, "a model for the prediction of thresholds, loudness, and partial loudness," *journal of the audio engineering society*, vol. 45, no. 4, pp. 224–240, april 1997.

[22] "Flucoma reference." [Online]. Available: https://learn.flucoma.org/reference/

[23] F. Itakura and S. Saito, "Analysis synthesis telephony based on the maximum likelihood method," *Proc. 6th of the International Congress on Acoustics*, 1968.

[24] C. Févotte, N. Bertin, and J. L. Durrieu, "Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis," *Neural Computation*, vol. 21, pp. 793–830, 3 2009. [Online]. Available: https://direct.mit.edu/neco/article/21/3/793/7395/Nonnegative-Matrix-Factorization-with-the-Itakura

[25] I. R. Bureau, "Requirements for loudness and true-peak indicating meters bs series broadcasting service (sound)." [Online]. Available: http://www.itu.int/ITU-R/go/patents/en

[26] "Learn more about ableton push | ableton." [Online]. Available: https://www.ableton.com/en/push/

[27] "Launch | novation." [Online]. Available: https://novationmusic.com/en/launch