Improving time travel experience by combining annotations

Adrien Vieilleribière
music@adrien-v.com

ABSTRACT

Since recorded audio material is played, navigating relevantly through it is a key expectation. This paper provides a formalism to introduce flexible navigation systems based on sets of annotations applying to the same audio object. It aims to build web interfaces to explore audio in time, robust for large data-sets and long files. Introducing the concept of weights applied to annotations, it specifies a parameterized version of the functionality **next**/**previous** and presents an effective implementation.

1. INTRODUCTION

In order to consume audio, interfaces need to deal with "Time Travel" *i.e.* to have some mechanism to control the current time[1]. For native html5 management of <audio> or in almost all web players, this task is performed using a slider[2] which generates two main problems:

Non relevant navigation. Choosing an arbitrary progression leads to position the play-head without any correlation with the audio content. Only having the "random" approach to explore audio files in time can be frustrating.

The big finger issue. As soon as the media is long and/or the display screen is small, it becomes challenging (sometimes impossible) to move the play-head by reasonable amounts of time. For instance, on a typical audio file of the corpus used for this paper, a forefinger has a size of several minutes on a phone screen and it's even worse on a watch screen.

This paper addresses those two limitations by exploring alternative interfaces dedicated to navigation in digital audio files. Advocated navigation scheme uses annotations corresponding to key spots in the file. It provides a formalism and interfaces to achieve discontinuous navigation strategies from a set of annotations.

[1]the property *currentTime* of <audio> in the html5 specification.
[2]*e.g.* <input type="range" min="0" max=length(audio) step="any">.

Web Audio Conference WAC-2016, April 4–6, 2016, Atlanta, USA

Outline. First, the experimental corpus (audio files and automatic annotations) is detailed and notations are fixed. Then, the corpus is equipped to be visualized. In section 4, controllers are built to interact globally on the relevant offsets. The deemed most relevant approach – the "Local Time Travel" – is detailed in section 5. It defines flexible **next** and **previous** functions[3] from a set of weighted annotations, and provides an interface to experiment them. The section 6 exhibits some methods to set weights by hand or by computation. Perspectives and conclusion end the article.

2. PRELIMINARIES

2.1 Corpus

The practical examples of the article are based on automatic annotations over the audio corpus of http://pul-lup.com

2.1.1 Audio files

The audio corpus manipulated is a set of 227 mixes of various styles: from zither to breakcore through jazz, hip hop, reggae .... Mix Lengths go from a couple of minutes to a few hours with an average of one hour, for a total of 10 days of audio contents.

2.1.2 Annotations

After audio corpus, annotation is the cornerstone of our navigation system, so let us build a corpus of annotations. For human production, we will focus on a generic description; and for automatic production on scalability.

Human annotations

In the majority of the modern audio editors, there is a dedicated format to store annotations. Even if many of those formats are very simple [11] or not documented [16] the idea is to enable users to create annotations in their favorite software. A generic xml schema is designed to unify the encoding[4] of an annotation and an import/export function is provided with several softwares including Audacity [11], Sonic Visualiser [16], AudioSculpt [12] ...

Algorithmic annotations

To obtain a significant quantity of annotations, several automatic segmenters were tried on the audio corpus and most of

[3]Inside the audio material, depending on the current time.
[4]Metadatas information about the annotation and the encoding of its segments.

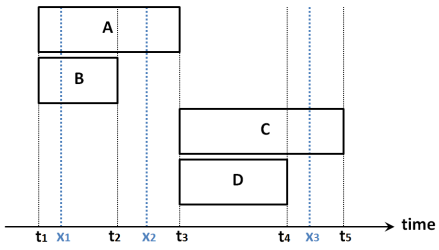


Figure 1: Example of 5 Segments: the topmost one is denoted $(t_1, t_3 - t_1, A)$.

them left aside: available implementations are often partial [9] or too difficult to access [6] and many of them [5] do not scale up for large audio files. The Queen Mary Segmenter [4, 15] was chosen and processed throught SonicAnnotator [1]. For each audio file, a few tens of segmentations were made by varying the three main parameters[5] of the segmenter, and a total of 15 000 segmentations were generated.

2.2 Notations

To formalize an annotation-based **next** and **previous**, let a be an audio of length T and Σ a finite alphabet. A *labeled segment* of a is a segment of $[0, T]$ labeled by a word on Σ. It will be denoted as a tupple $s = (t, d, w)$ with $0 \leq t \leq T$, $0 \leq d \leq T - t$, and $w \in \Sigma^*$. t corresponds to the starting time; d corresponds to the duration; and w to the label of the labeled segment.
An *annotation* A of size n of an audio file a is a set $\mathcal{A}$ of n labeled segments of a and a set $\mathcal{M}$ of metadatas.
Formally, for an annotation A, the $\mathcal{N}ext$ operation at time t gives the set of the closest subsequent labeled segments and the *next* is the (common) starting time of these segments:

- $\mathcal{N}ext_A(t) = \{ (t_i, d_i, w_i) \in \mathcal{A}$ such that $t_i > t$ and
$$\forall j \in \{1, n\}, (t_i > t) \Rightarrow (t_j \geq t_i) \}.$$
- $next_A(t) = T$ if $\mathcal{N}ext_A(t)$ is empty; and $next_A(t) = t_i$ such that $(t_i, d_i, w_i) \in \mathcal{N}ext_A(t)$ otherwise.

Symmetrically:

- $\mathcal{P}rev_A(t) = \{ (t_i, d_i, w_i) \in \mathcal{A}$ such that $t_i < t$ and
$$\forall j \in \{1, n\}, (t_i < t) \Rightarrow (t_j \leq t_i) \}.$$
- $prev_A(t) = 0$ if $\mathcal{P}rev_A(t)$ is empty; and $prev_A(t) = t_i$ such that $(t_i, d_i, w_i) \in \mathcal{P}rev_A(t)$ otherwise.

Example 1. For the annotation A with segments given by the Figure 1, and summarizing the segment by its label: $\mathcal{N}ext_A(x_1) = \{C, D\}$, $next_A(x_1) = t_3$, $\mathcal{N}ext_A(x_3) = \emptyset$, $next_A(x_3) = T$, $\mathcal{P}rev_A(x_1) = \mathcal{P}rev_A(x_2) = \{A, B\}$, $prev_A(x_1) = prev_A(x_2) = t_1$.

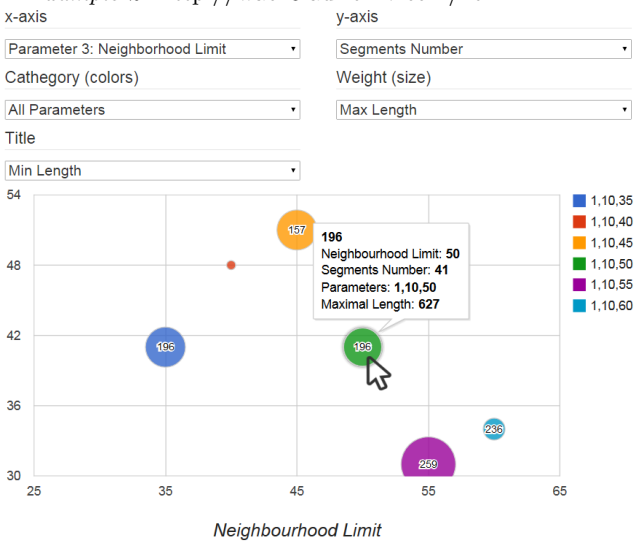
3. VISUALIZATION: SCALAR VALUES

To explore this huge generated set of annotations, we first construct a data cube of quantitative properties of the generated segmentations (average length of segments, lengths of the shortest and the longest one . . .) and then use computational and visualization tools. For large datasets, statistical analysis is used and for smaller ones, a prototype of interactive visualization is built.

Small data set

A web interface is built to visualize the cube of data. The user chooses the mapping between the available properties of annotations (11 dimensions including parameters used, average duration of segments, longest duration. . .) to the 4 dimensions[6] (+title) of a bubble chart visualization [14].

Example 2. http://wac16.adrien-v.com/?ex=2



In this example, each bubble represents a segmentation of the same audio file. The parameters used for the bubble are here shown as the category: hybrid features are used (first parameter), 10 segment types are searched (second one), and the neighborhood limit (third one) is shown in x-axis. It shows that raising the neighborhood limit does not always decrease the number of found segments. The size of the bubbles is mapped to the maximal duration: it points out that the four larger bubbles have some large uncut portions (over 10 minutes).

Large data set

A more sophisticated approach is needed to obtain readable plots from large data sets. Thus, to analyze the whole cube of our corpus, data is injected into a statistical system [7, 2] and correlations are projected with a Principal Component Analysis [3]. A few examples of such projections can be found at http://wac16.adrien-v.com/PCA.php.

4. GLOBAL TIME TRAVEL

We want a framework to navigate with several annotations and now focus on time related data and interfaces to navigate through time.

To visualize a set of segmentations and use them for interaction, each segment is associated to a visual object that can access its attributes[7].

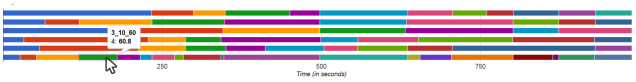
[5]The features used (Chroma,Constant-Q,MFCC), the Number of Segment Types, and the Neighborhood limit.

[6]x-axis, y-axis, weight (size of the bubbles) and category (color).

[7]Starting position, duration and label.

We can use a standard Bar Charts [13] and change the reading position on selected events as in the following example.

Example 3. http://wac16.adrien-v.com/?ex=3



Time goes from left to right, each line represents a segmentation, and each rectangle a segment. The focus shows the parameters used for the line and the length (in seconds) of the segment. Clicking on a segment puts the reading position into its starting time.

This type of controller succeeds in replacing the random click search method, which solves the first "relevant navigation" expectation. However, for long audio files, the "big finger issue" remains. One can go through this limitation with a zoom capability, for instance by the use of annotation charts [10] (e.g. http://wac16.adrien-v.com/?ex=4).

This answers both problems addressed in the introduction. It creates new frustrations: the zoom capability is hard to be quickly set to an appropriate granularity, and hard to update without degrading the user experience.

5. LOCAL TIME TRAVEL

To get around previous annoyances, we suggest a local navigation strategy. It can be understood as an extended concept of the **next** or **previous** function based on a set of annotations. All the annotations may contribute to this function but not necessary with an equal significance. First of all, let us assume that a weight $\lambda_i \geq 0$ is set on each annotation A_i.

Minimal Local Navigation

The simplest navigation with several annotations is using the union of the segments:

$$next^{\lambda_1,\ldots,\lambda_n}_{A_1,\ldots,A_n}(t) = min^n_{i=1}\,(next_{A_i}(t) \text{ such as } \lambda_i > 0)$$

This navigation can be very useful for tasks requiring all possible hot points. However, it can easily create very small segments and will not provide an interesting user experience in most cases.

Maximal Local Navigation

To navigate with bigger shifting, we can think of going forward until every annotation has started a new segment:

$$next^{\lambda_1,\ldots,\lambda_n}_{A_1,\ldots,A_n}(t) = max^n_{i=1}\,(next_{A_i}(t) \text{ such as } \lambda_i > 0)$$

At the opposite of the previous case, the navigation experience can be downgraded by too large shifts due to sparse annotations.

General Case: Parameterized Local Navigation

The most interesting **next** functions are "between" those minimal and maximal cases. To get such intermediary functions, a parameterized version is proposed: the α-**next** navigation[8]. The intuition is that the parameter α is related to "how far" the user wants to go forward (e.g. Figure 2).

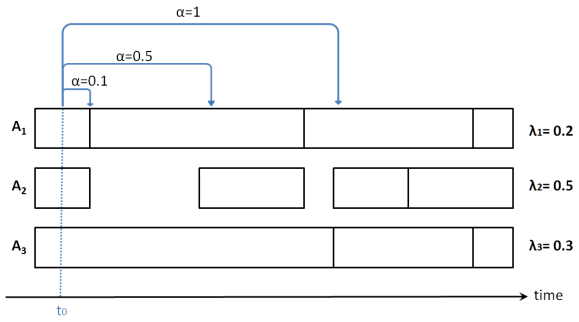


Figure 2: Examples of "weighted nexts".

More formally, let $\mathbb{A} = (A_1, \ldots, A_n)$ be a set of n annotations of the audio file a with weights $\lambda = (\lambda_1, \ldots \lambda_n), \lambda_i \geq 0$ and let $\alpha > 0$ be a target weight.

$\mathcal{N}ext^{\lambda}_{\mathbb{A}}[\alpha](t)$ is the set[9] of $(t_i, d_i, w_i) \in \mathcal{A}_j$ such that

- $\lambda_j > 0$,
- $t_i > t$,
- $\alpha \leq \sum_{j=1}^{n} \lambda_j \delta_{next_{A_j}(t) \leq t_i}$ and
- for all $(t', d', w') \in \mathcal{A}_{j'}$ with the first three properties, $t' \geq t_i$.

As before, the $next^{\lambda}_{\mathbb{A}}[\alpha](t)$ is defined as T if $\mathcal{N}ext^{\lambda}_{\mathbb{A}}[\alpha](t)$ is empty and as the starting time of the segment(s) returned by $\mathcal{N}ext^{\lambda}_{\mathbb{A}}[\alpha](t)$ otherwise.

Remarks: The minimal case naturally extends the definition for $\alpha = 0$. To redefine the maximal case, just set α to $\sum_i \lambda_i$. Computing the $next(t)$ can be done accumulating weight in a loop reading all annotations in parallel. It can be simplified and optimized[10] for both minimal and maximal cases.

Let's play it online

We have a corpus of segmentations, and an algorithm to compute the weighted **next** or **previous** function. It is time to build a web interface to experiment the weighted local time travel. An example – shown in Figure 3 – can be manipulated at http://wac16.adrien-v.com/?ex=wltt.

6. SETTING WEIGHTS

From a general point of view, setting a weight to each annotation can be used for many purposes:
- one can simulate a filter mechanism using binary weights.
- scalar weights can represent a human ranking system.
- weights can also model an arbitrary computation of relevance ...

[8]The α-**previous** definition is symmetric.

[9]For $j \in \{1, n\}$ and j_i from 1 to the size of $\mathcal{A}_j$.

[10]The minimal case can be computed in a simple loop over all the segments of all the annotations and the maximal case can be treated with a loop over annotations containing a loop over its segments.

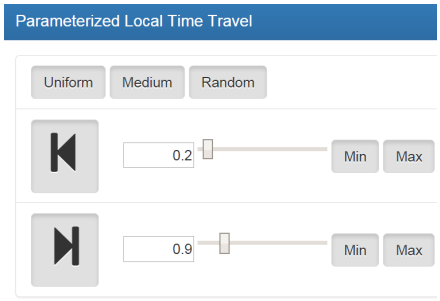


Figure 3: Weighted Local Time Travel Controler.

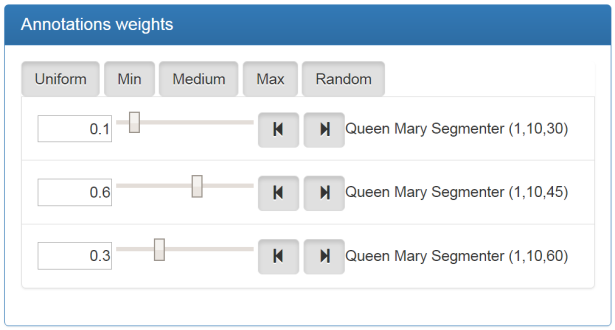


Figure 4: A user interface to set weights λ_j.

In practice, for human driven ranking systems, a simple interface is built to allow a user to choose its own weights (Figure 4).

For an automatic approach, many heuristics can be designed for specific requirements (the distance to an ideal average length, functions of the maximal length of a segment...). To have a more neutral approach, the weight of an annotation can be dynamically computed according to its statistical correlation to the other annotations over the same audio file (e.g. http://wac16.adrien-v.com/weightsPCA.php).

- First we compute the distances d_i to the center of the Principal Component Analysis.
- Then we "reverse" these distances to enhance the central points: $x_i = f(d_i)$, with f a decreasing positive function.
- And finally the total value is normalized to 1: $\lambda_i = \frac{x_i}{\sum_i x_i}$.

7. PERSPECTIVES AND FUTURE WORK

For clarity's sake, the cube of section 3 is quite simple but could be adapted to any data values with arbitrary dimensions. In a future study, we would like to use the same principles to quantify the relevance of various extracted features for time travel.

The interface to set weights will be generalized in order to integrate social indicators (likes, number of usages, finer ranking systems ...) and to implement the statistical approach of section 6.

The corpus of annotations would be more interesting with data from humans and from other segmenters [8, 6]. The next step is to compare several segmenters to track how to minimize distances from human annotations.

8. CONCLUSION

The paper presents various solutions to "Time Travel" through an audio file from a set of weighted annotations. Web visualization and statistical analysis are used to scroll the annotations and set weights. In this framework, the user gets a parameterized functionality **next/previous**. The suggested mechanism can deal with both long audio files and large datasets of annotations, and Relevant navigation is achieved: only interesting times are targets of the playhead controlled by the user.

9. REFERENCES

[1] C. Cannam, M. O. Jewell, C. Rhodes, M. Sandler, and M. d'Inverno. Linked data and you: Bringing music research software into the semantic web. *Journal of New Music Research*, 39(4):313–325, 2010.

[2] F. Husson, J. Josse, S. Le, and J. Mazet. *FactoMineR: Multivariate Exploratory Data Analysis and Data Mining*, 2015. R package version 1.31.3.

[3] I. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 2002.

[4] M. Levy and M. Sandler. Structural segmentation of musical audio by constrained clustering. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(2):318–326, Feb 2008.

[5] M. Mauch. *Automatic Chord Transcription from Audio Using Computational Models of Musical Context*. PhD thesis, Queen Mary, University of London, 2010. Features a more thorough description of the segmentation algorithm in Chapter 6.

[6] G. Peeters. Mirex 2009 "music structure segmentation" task: Ircamsummary submission.

[7] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015.

[8] I. Theodorakopoulos, G. Economou, and S. Fotopoulos. Unsupervised music segmentation via multi-scale processing of compressive features' representation. In *Digital Signal Processing (DSP), 2013 18th International Conference on*, pages 1–6, July 2013.

[9] D. R. Turnbull. *Design and development of a semantic music discovery engine*. UC San Diego: b6635969, 2008.

[10] Annotation chart: https://developers.google.com/chart/interactive/docs/gallery/annotationchart.

[11] Audacity: http://audacityteam.org/.

[12] Audiosculpt: http://forumnet.ircam.fr/fr/produit/audiosculpt/.

[13] Bar charts: https://developers.google.com/chart/interactive/docs/gallery/barchart.

[14] Bubble chart: https://developers.google.com/chart/interactive/docs/gallery/bubblechart.

[15] Queen mary segmenter vamp plugin: http://vamp-plugins.org/plugin-doc/qm-vamp-plugins.html#qm-segmenter.

[16] Sonic visualiser: http://www.sonicvisualiser.org/.