Audio inpainting with similarity graphs

Nathanael Perraudin, Nicki Holighaus, Piotr Majdak, Peter Balazs

***Abstract*—In this contribution, we present a method to compensate for long duration data gaps in audio signals, in particular music. To achieve this task, a similarity graph is constructed, based on a short-time Fourier analysis of reliable signal segments, e.g. the uncorrupted remainder of the music piece, and the temporal regions adjacent to the unreliable section of the signal. A suitable candidate segment is then selected through an optimization scheme and smoothly inserted into the gap.**

I. Introduction

The loss or corruption of data segments of considerable duration is a very common issue in data restoration and transmission. In audio applications in particular, the insertion of *perceptually pleasing* content is very important. A good insertion would prevent audible artifacts and provide a coherent and meaningful signal to the listener who would, optimally, remain unaware that any problem has occurred. This task has recently become known as audio inpainting [1], but has previously been referred to e.g. as audio interpolation [2] or waveform substitution [3]. Audio inpainting aims at reconstructing missing parts of an audio signal. When missing parts have a length of a few samples, sparsity based techniques can be used [1], [4], [5]. However, these algorithms are not able to provide satisfactory results for distortions longer than 50ms. For such cases, techniques relying on autoregressive modeling [2], sinusoidal modeling [6], [7] or based on self-content [8] have been proposed with varying degrees of success, depending on suitable assumptions on the signal at hand.

Here, we propose a novel method that can be considered a contribution to the latest category. Although the algorithm provided can be applied to any (audio) signal, it is implicitly assumed that for any short signal segment, we can find another segment in the given reliable data with similar spectro-temporal behavior. It is easy to see that this condition is satisfied for various musical genres such as pop and rock music with a somewhat predictable structure. For example, if a segment of a few seconds duration is missing from a pop song, then with high probability there is a another segment in the signal with very similar content. However, even for less predictable signals such as certain classical or jazz music pieces, it is often possible to find a highly suitable match. Given a defective signal segment and a set of reliable content, usually the uncorrupted remainder of the music piece, our method finds such a match and automatically aligns it with the defective segment, providing a restored version of the signal. Figure 1 illustrates how this is equivalent as jumping to a different part of the song during the gap. Thus a natural replacement needs to have 2 "ear-friendly" transitions denoted T_1 and T_2.

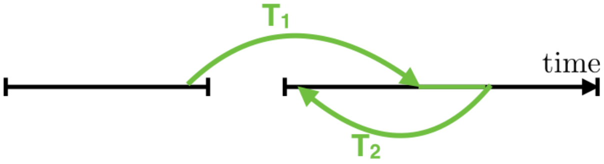

Figure 1. The basic idea of segment-based audio in-painting. In order to replace the missing segment, the algorithm uses a different part of the signal. T_1 and T_2 are the resulting transitions.

Even though music is in general redundant, there are segments that appear only once during a song. If the missing part lays in the middle of a unique segment, it is not possible to find a replacement matching exactly the missing part. In order to cope with this problem, we allow ourselves to replace some of the known content and to modify slightly the signal length. Thus, as illustrated in Figure 2, our algorithm searches for a replacement segment that optimally satisfies the 3 following criteria:

1) The transitions T_1 and T_2 (green dashed lines in Figure 2) resulting from the pasting operation should be as imperceptible as possible. Ideally, the listener should not be able to notice the transition, even if the inpainted content does not equal the missing data.
2) The lengths of the replaced reliable, known information L_1 and L_2 should be as small as possible.
3) The length of the song should remain approximately the same, i.e: the length of the replacement D_2 should be close to the length of the replaced content D_1.

Some margin for compromise is, however, essential to the construction of a good solution. Since the question of how strictly the reliable content is to be preserved is highly application-dependent, a parameter in the optimization scheme enables the tuning of this property.

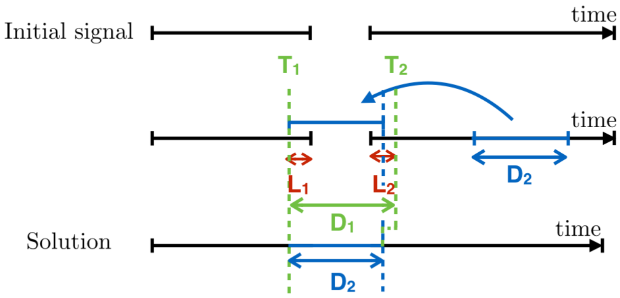

Figure 2. What is the best replacement segment to replace the missing content? The transitions T_1 and T_2 should not be perceptible. The length of the replacement D_2 should be close to the length of the original segment D_1. Known, reliable information should be overwritten as little as possible, i.e. L_1 and L_2 should be small.

An exhaustive search for possible replacements is intractable as the number of candidate segments grows about quadratically in the signal length, which can easily be millions of samples. The main idea is to construct a small set of matches which all have a sufficiently good transitions quality. In order to assess this quality, we construct a similarity graph based on spectro-temporal information obtained through a short-time Fourier analysis [9], [10] of the reliable content and the signal regions adjacent to the *gap* (the defective segment). More specifically, our time-frequency features are given by time-weighted, connected patches of short-time Fourier coefficients, respectively their amplitudes and partial phase derivatives in the time direction, see Section III-B for more details. Once the set of possible candidates is built, the optimal match is determined through an optimization balancing the difference of the time-frequency features and the length difference of the defective and candidate segments. The final inpainting procedure itself is then simply a cross-faded transition between the original border regions around the gap and the determined match, as illustrated by Figure 1.

A. Related Work

The audio inpainting problem has mainly been addressed from a sparsity point of view. The hypothesis is that usual sounds are composed only with a few time-frequency atoms. Using classical ℓ_0 or ℓ_1 optimization techniques, algorithms have been designed to inpaint short sound gaps [1], [4]. Audio inpainting is known as "waveform substitution" [3] by the community addressing packet loss recovery techniques [11].

More related to this contribution, similarity-based audio inpainting has already been proposed in [8]. Similarly to our own contribution, the authors design an algorithm searching for similar parts of the signal using time-evolving features. However, the approach developed in [8] is designed to handle small holes (up to 40 ms) of information originating from packets being dropped during transmission. The resulting algorithm is fairly different as it 1) does not use a similarity graph, 2) does not allow replacement of known content, 3) uses a probabilistic prior along the similarity measure and 4) has a different objective function as its goal is not to fool the listener but to restore the lost content.

Finally, the idea that the music is highly structured with deliberate similarities is not new and has been already explored [12], [13]. The work presented in these contributions paved the way for founding "The Echo Nest"[1], a company specialized into audio feature design. They have already build graphs of similarities for the infinite jukebox: http://labs.echonest.com/Uploader/index.html

B. Contributions

In this contribution we propose a method to build a similarity graph for audio signals (see Section III-C). As explained in Section II, this graph encodes an important part of the structure of the audio signal. Based on this structure, we propose in Section III an algorithm to solve audio inpainting problems. By finding the *most similar* segment from a body of candidate data, the resulting method is able to inpaint large gaps inside the audio signal, without a loss in audio quality and largely independent of the complexity of the signal at hand. Finally, in Section IV, we perform a preliminary evaluation of the algorithm performance, computationally and with regards to perceptual quality.

II. A TRANSITION GRAPH ENCODING MUSIC STRUCTURES

The problem we consider, i.e. how to restore a piece of music when an extended, connected piece has been lost or corrupted, often requires us to abandon the idea of exact recovery. In the case where only a short segment (up to about 50ms) has been lost [1], or the signal can be described by a very simple structure [6], it may be possible to infer the missing information from the regions directly adjacent to the distortion with sufficient quality. However, for complex music signals and corruptions of longer duration, such inference remains out of reach. Hence, we instead ask the question: *What characterizes a music piece, what makes it sound natural, as opposed noise or other environmental sounds?*

Of course, the answer to this question is not only highly subjective, but also content-dependent. As much as different genres adhere to differing rules, so do different listeners reactions to music vary, depending on the listening experience and habits. A listener accustomed to classical, or jazz music will have her attention drawn by different musical cues than one mostly familiar with pop or rock. Nevertheless, the often (though surely not always) predictable evolution of distinct and structured rhythmic and/or harmonic patterns plays an important role in the experience and recognition of music. Everyone has experienced the instant recollection of a whole song from a pattern as short as a few notes or beats.

But even if a pattern is not repeated in the exactly same fashion, the conscious variation of previous structures, rhythmic, harmonic or otherwise, is an integral part of music, although the grade of self-similarity inside a single piece of music can greatly vary. These patterns and their temporal development are what provides coherence and structure to a piece of music.

Going back to the original problem of music restoration, it seems natural to exploit this type of *redundancy* in the musical piece to be restored. The analysis of a piece of music based on this these considerations can be abstracted into determining the temporal evolution of spectral content in the signal. Clearly, this is an oversimplification and many effects related to human auditory perception and the processing of sound data in the human brain play an important role, but *simple* time-frequency analysis can provide a surprisingly suitable first approximation.

Inspired by this observation, we construct an audio similarity graph. The vertices of the graph represent small parts of musical content, while the edges indicate the similarity between the segments in terms of local spectral content. The crucial step towards good performance is once more the enforcement of temporal coherence, i.e. respecting the time-dependent structure of the underlying music. This is achieved by emphasizing such connections in the graph that persist over some span of time.

[1] http://the.echonest.com/

Besides providing an intuitive analysis of the signal at hand, exposing self-similarities and global structure, it can be used for a number of different purposes, e.g. re-compose a song by following the edges of the graph, respecting the global music structure[2]. In this contribution, we use the similarity graph to address the problem of audio inpainting. Leveraging the information contained into the graph, we design an algorithm able to replace some content inside the music, respecting the coherence and structure of the underlying music piece.

III. Method

In this section, we describe the proposed inpainting method. We begin with an overview of the algorithm, before discussing the individual stages in more detail.

A. Stages of the Algorithm

The presented algorithm attempts the restoration of a defect in a one channel audio signal, usually given the entire (possibly corrupted) audio signal and the position of the part to be replaced. In particular, we consider the problem of dealing with long duration corruptions in the range from form 0.1s to a few seconds. Simply put, our algorithm selects, according to some numerical criterion, the optimally similar segment from the reliable part of the signal and inserts it into the gap. Our framework is easily adapted to multi-channels signals, by applying the inpainting to all channels simultaneously.

The aim of the algorithm is to provide autonomously a restored signal that sounds as natural as possible. That is, we prefer a less accurate[3] result with little artifacts over a more accurate solution with audible artifacts. This is reflected in the algorithm mainly grading the transitions themselves instead of the *inner part* of the inpainted segment. However, this also implies that a proper evaluation of the algorithm performance cannot be based on a numerically motivated, objective difference measure, but is more involved. This issue is touched upon in Section IV. On the other hand, in the context of real world applications, the missing original signal segment is unknown and cannot usually be recovered, while a measure to judge the transition quality still provides a crucial first estimate of the reliability of the restoration.

As alluded to in the previous paragraph, the main task of our method is the determination of a set of segments with suitable natural transitions T_1 and T_2, recall Figure 2. The steps in which that is achieved are summarized in Algorithm 1. The rationale behind the algorithm can be described as follows: 1) Create feature vectors that represent the audio content at a specific time. The difference of feature vectors at different time positions quantifies the quality of a transition between these two positions. 2) In order to simplify our analysis, construct a graph in which sufficiently similar positions are connected by an edge. The edge weight depends on the difference of the corresponding feature vector, i.e. they represent *good* transitions between time points. 3) Select the optimal segment, considering the transition quality T_1 and T_2 and the difference between the segment lengths D_1 and D_2, see Figure 2. 4) Eventually, a restored signal is synthesized by inserting the selected similarity based content instead of the corrupted part.

Algorithm 1 Similarity-based audio inpainting algorithm

```
1: Local audio feature extraction (Detailed in subsection III-B)
2: Graph creation (Detailed in subsection III-B)
3: Transition selection (Detailed in subsection III-D)
4: Signal reconstruction (Detailed in subsection III-E)
```

B. Local audio features

The audio feature selection is crucial for the reliability and efficiency of our algorithm. The subsequent steps depend on the given similarity rating, i.e. the distance in terms of the features, to imply a good match. In this contribution, we focus on features that can be obtained from a sampled short-time Fourier transform (STFT) of the signal. The STFT of a signal $s \in \mathbb{C}^L$ with respect to a window function $g \in \mathbb{R}^L$ is defined as

$$V_g s[n,m] = \sum_{l=0}^{L-1} s[l]g[l-n]e^{-2\pi iml/L} = \widehat{s \cdot g[\cdot - n]}[m],$$

where the vector $V_g s[n,\cdot]$ measures the local frequency content of s around n. Since we assume $s \in \mathbb{R}^L$, the vectors $V_g s[n,\cdot]$ are Hermitian symmetric and it is sufficient to consider $m = 0,\ldots,\lfloor L/2 \rfloor + 1$. If only the values $V_g s[na, mL/M]$, for some divisors $a, M \in \mathbb{N}$ of L, are taken, we say that a sampled STFT with hop size a and M channels is computed.

Data preprocessing. Before the features are computed, the signal might be down-sampled if the original sampling rate ξ_s is higher than $\xi_{s,\max} = 12$ kHz. Signals are down-sampled by a factor $\lceil \xi_s/\xi_{s,\max} \rceil$ after applying an anti-aliasing filter. Besides a convenient complexity reduction, this also restricts the audio feature to a smaller frequency range that still contains the most relevant information.

Of several features directly derived from the sampled short-time Fourier transform[4], a feature based on the dB-spectrogram, combined with partial phase derivatives in the time direction, has produced the most reliable results. Given the STFT parameters g, a, M, we first compute the STFT of s with respect to g, with hop size a and M channels, obtaining a matrix $\mathbf{C} \in \mathbb{C}^{M \times L/a}$, where $\mathbf{C}_{m,n} = V_g s[na, mL/M]$. The local features are obtained in two steps.

dB-spectrogram feature. The human auditory system percieves loudness approximately as a logarithmic function of sound pressure. Therefore, the db-spectrogram provides natural time dependent audio features. It is defined as

$$\mathbf{S}^{\mathrm{db}}_{m,n} = 20 \log_{10}(|\mathbf{C}_{m,n}|).$$

[2] A similar method has been used for that exact purpose in the so-called “infinite jukebox”, available at http://labs.echonest.com/Uploader/index.html.

[3] According to a numerical quality function such as SNR of the inpainted segment versus the unknown original.

[4] The candidates were: spectrogram, dB-scaled spectrogram, MFCC [14], partial phase derivative in time-direction and combinations thereof

In order to avoid the large negative values that are obtained if $\mathbf{C}_{m,n}$ is small, we modify $\mathbf{S}^{\text{db}}$ as follows, to obtain the first feature set.

$$\mathbf{F}^{(1)}_{m,n} = p^{-1}\left(\mathbf{S}^{\text{db}}_{m,n} - \max_{k,l}(\mathbf{S}^{\text{db}}_{k,l}) + p\right)_{+},$$

where $(x)_{+} = x$, if $x > 0$, and 0 otherwise. Here, p is a parameter given in dB that specifies the dynamic range that is considered, and the normalization with p^{-1} guarantees that we obtain values in $[0, 1]$. By default, we set $p = 50$.

Phase derivative feature. The fact that the partial derivative in time direction of the phase of the STFT provides a good estimate of instantaneous frequency has been used extensively in the literature. In particular, it is used in the popular reassignment and synchro-squeezing methods for spectrogram deconvolution [15], [16]. For our purposes it is interesting that the time-direction phase partial derivative attains large values mostly in the vicinity of sustained sinusoidal signal components. Moreover, its magnitude is independent of the energy in that component, but closely related to the distance to the instantaneous frequency. Therefore, the phase partial derivative can serve to put additional emphasis on sinusoidal signal components. The phase of the STFT can be obtained as

$$\phi[m, n] = (2\pi)^{-1}\log\left(\frac{\mathbf{C}_{m,n}}{|\mathbf{C}_{m,n}|}\right).$$

and a discrete derivative can be obtained in each fixed channel m by a finite difference scheme operating over n. However, Auger and Flandrin [15] have shown that the desired phase derivative can be obtained as $-\mathbf{Im}(V_{g'}s[n, m]/V_g s[n, m])$, where g' is a suitable discrete derivative of the window function g. This is usually more efficient and accurate than computing a finite difference scheme, but most importantly, it can be computed without accuracy loss in the presence of arbitrary hop sizes a and number of channels M. To that purpose, we define $\mathbf{C}^{td}_{m,n} = V_{g'}s[na, mL/M]$ and

$$\mathbf{P}^{td}_{m,n} = -\mathbf{Im}(\mathbf{C}^{td}_{m,n}/\mathbf{C}_{m,n}).$$

We are mainly interested in the phase derivative of high energy components in the spectrogram. Therefore, we multiply $\mathbf{P}^{td}$ with a binary mask

$$\mathbf{M}_{m,n} = \begin{cases} 1 & \text{if } |\mathbf{C}_{m,n}| \geq c_{\text{thr}} \max_{k,l}(\mathbf{C}_{k,l}), \\ 0 & \text{else,} \end{cases}$$

where c_{thr} is a threshold parameter, set by default to $c_{\text{thr}} = 0.002$. Finally, the phase derivative is smoothened by convolution with a localized kernel, suppressing quick changes. The second feature set is thus obtained by

$$\begin{aligned}\mathbf{F}^{(2)}_{m,n} &= \tilde{\mathbf{F}}^{(2)}_{m,n}/\max_{k,l}(\tilde{\mathbf{F}}^{(2)}_{k,l}), \qquad \text{with } \tilde{\mathbf{F}}^{(2)}_{m,n} \\ &= \left(\mathbf{M}_{m,\cdot}\mathbf{P}^{td}_{m,\cdot}\right) * v_{\text{ker}}[n].\end{aligned}$$

Here, $*$ denotes circular convolution and v_{ker} is a positive, symmetric convolution kernel, centered at $n = 0$. The default convolution kernel is an 8-point Hann window.

The resulting features. The two feature types can be weighted against one another with a parameter λ, resulting in the local audio features

$$\mathbf{F}_n = (\mathbf{F}^{(1)}_{1,n}, \ldots, \mathbf{F}^{(1)}_{M-1,n}, \lambda\mathbf{F}^{(2)}_{1,n}, \ldots, \lambda\mathbf{F}^{(2)}_{M-1,n})^T, \quad \text{for } n = 0, \ldots, L/a - 1.$$

$\lambda = 3/2$ has proven to be a good default value. In the next step, we construct a similarity graph by computing pairwise norm differences between the local audio features.

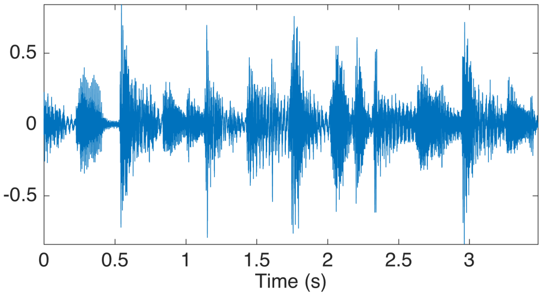

Figure 3. Time-domain signal. The waveform representation reveals only little about the signal structure and is not suitable for a similarity analysis.

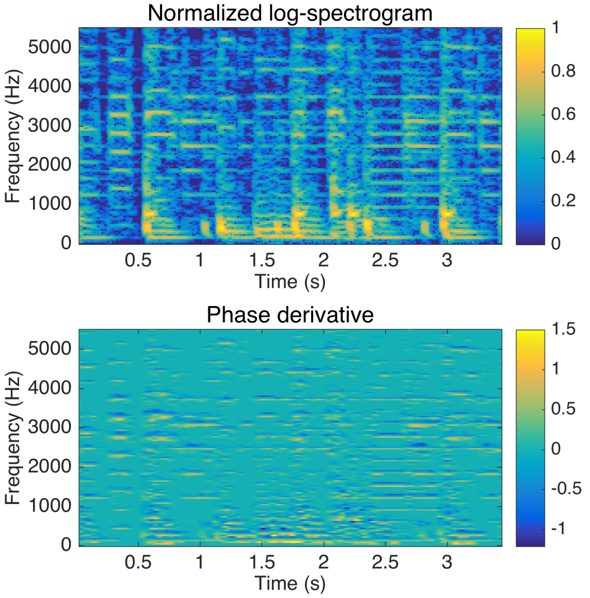

Figure 4. Time-frequency representations of the signal. The features shown are based on the log-spectrogram (top) and a partial phase derivative (bottom).

C. *Creation of the transitions graph*

The transition graph is an essential tool in our analysis. It contains the potential transitions that the algorithm considers as sufficiently good inside the analyzed signal content. The edge weights rate the quality of the transitions in terms of the features considered in the previous section. The graph contains L/a nodes which represent the possible time instants for the transitions. Its construction requires some care and is done in two steps. First we build a traditional nearest neighbors graph G_0 with weight W_0. To build the final graph

G with weight W, we then apply some refinements that ensure persistence of the transition candidates in time and a sufficiently sparse graph only containing the most relevant connections.

Find the nearest neighbors. For all feature vectors we search for the ℓ_2-norm k nearest neighbors. Since this operation is expensive, we use the FLANN library (Fast Library for Approximate Nearest Neighbors) [17] to efficiently provide an approximate solution. Following a traditional graph construction scheme, we then associate a preliminary weight to every connection. Let us write k_i the set of the k (approximate) nearest neighbors of the the vector x_i. The weights of the initial graph are computed with a standard Gaussian kernel and given by:

$$W_0(i,j) = \begin{cases} e^{-\frac{\|x_i - x_j\|_2^2}{\sigma}} & \text{if } j \in k_i \\ 0 & \text{if } j \notin k_i, \end{cases}$$

where x_i and x_j are two feature vectors. σ is set to the average squared nearest neighbor distance

$$\sigma = \frac{1}{|I|k} \sum_{i \in I} \sum_{j \in k_i} \|x_i - x_j\|_2^2,$$

where I is the feature index set, i.e. the set of all vertices of the graph. If $\|x_i - x_j\|_2^2 = d_{ij}^2 \ll \sigma$, i.e. the feature vectors are similar, the weight $W_0(i,j)$ will be approximately 1. On the contrary, when the distance is larger, the weight approaches 0. Figure 5 left shows the weight matrix W_0. The diagonal shape of the non-zero elements indicates that the audio signal has similar segments persistent through several (time-)adjacent feature vectors. This property is desirable for a good transition because it indicates that the similarity is consistent along time, for at least a short duration. Therefore, the next step aims to emphasize such connections, while discarding instant similarities without persistence over time.

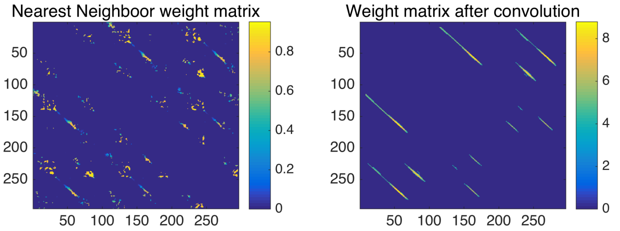

Figure 5. Left: nearest neighbors weight matrix. Right: Weight matrix after convolution and hard-thresholding.

Enhance time-persistent similarity. In order to extract transitions that are consistent along time, we enhance these diagonal effects by convolving the weight matrix with a diagonal kernel shown in Figure 6. The length of this kernel is defined by the variable l_k. This operation considers l_k consecutive feature vectors, implying that the algorithm analyses signal segments of length $\frac{a l_k}{f_f}$. With the default parameters of Table V, this equals approximately half a second. The convolved weight matrix reads:

$$W_c(i,j) = \sum_{u=-l_k/2}^{l_k/2} c_u W_0(i+u, j+u),$$

where $c_u = \frac{l_k - 2|u|}{l_k}$. The kernel c is shown in Figure 6.

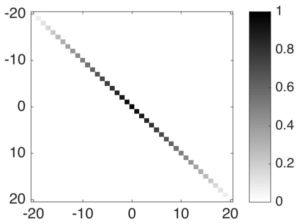

Figure 6. Convolution kernel used to enhance the diagonal shape of the weight matrix. Here $l_k = 40$.

Thin out the graph. For the purpose of signal restoration, we are only interested in the strongest connections, i.e. those with weights close to 1. Small weights are eliminated by a simple hard thresholding with threshold t_w, to reduce the number of connections in the graphs and thereby the computational load for selecting the best connections see Section III-D. The result is shown in Figure 5(r). An example for the resulting graph, displayed in Figure 7(l), may still contain a large number of connections, possibly pointing to the same similar segment in the signal. In order to overcome this problem, the number of connections is further reduced, by only selecting the local maxima in the final weight matrix, i.e. after the convolution step. The final weight matrix is thus

$$W(i,j) = \begin{cases} W_c(i,j) & \text{if } \begin{cases} W_c(i,j) \geq t_w \\ W_c(i,j) \geq W_c(i+1,j+1) \\ W_c(i,j) \geq W_c(i-1,j-1) \\ W_c(i,j) \geq W_c(i+1,j-1) \\ W_c(i,j) \geq W_c(i-1,j+1) \end{cases} \\ 0 & \text{otherwise} \end{cases}$$

The resulting sparse graph is shown in Figure 7(r).

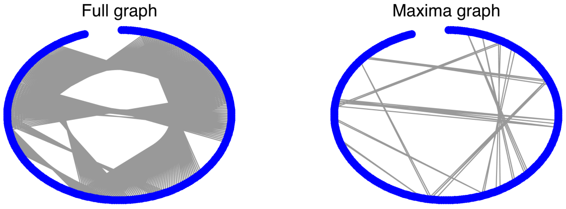

Figure 7. Left: graph with all possible transitions, i.e: $W_c(i,j) \geq t_w$. Right: sparsified graph by taking the maximum connections.

Reduce computational cost. For our specific problem, only a partial transition graph needs to be computed. In particular, we are only interested in outgoing connections in a short region, i.e. in the range of few seconds, immediately before the corrupted segment and incoming connections in a similar region immediately after the gap. Conceptually, we want the values of L_1 and L_2 in Figure 2 to be small. Therefore, the k nearest neighbors search is not performed on all $i \in I$, but only for a small subset of vertices in the direct vicinity of the corrupted segment. In practice, we limit ourselves to 5 seconds before and after the gap. Figure 8 shows an example of the resulting graph.

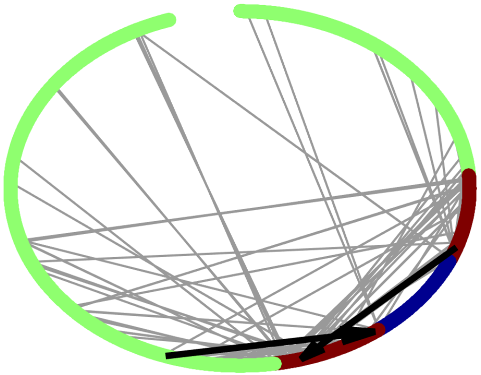

Figure 8. Subgraph. Blue: hole. Red node used for the subgraph. The two black connections are the transitions T_1 and T_2.

D. Selection of optimal transitions

At this stage, the connections in the graph represent the best connections between the border regions around the gap and the reliable data, graded with respect to the described feature vectors. In order to provide a satisfactory inpainting result, we require, however, a pair of transitions with the following properties:

1) Known content is overwritten as little as possible, i.e. L_1 and L_2 in Figure 2 should be minimized.
2) The individual transitions should have optimal quality, i.e. T_1 and T_2 in Figure 2 should have a very good rating. The rationale here is that a transition with a good grade will in most cases also be perceptually pleasing.
3) The final length of the reconstructed signal does not diverge much from the original signal length, i.e. the difference between D_1 and D_2 in Figure 2 is to be minimized.

In order to trade-off between these requirements, we formulate an optimization problem:

$$\begin{aligned}\operatorname*{arg\,min}_{\{T_1,T_2\}\in S} \quad & |D_1(T_1,T_2) - D_2(T_1,T_2)| \\ + \quad & \gamma_1 \left(L_1(T_1) + L_2(T_2)\right) \\ + \quad & \gamma_2 \left(\frac{1}{w(T_1)} + \frac{1}{w(T_2)}\right)\end{aligned}$$

where γ_1 and γ_2 are the regularization parameters balancing between the importance of the different constraints and S the set of valid transitions. This set contains all pairs of transitions obtained from the graph that also satisfy the following two properties: (a) the transition T_2 is starting after the end of T_1. (b) The of T_1 and the beginning of T_2 are on the same side of the gap. The transitions are displayed as gray arrows in Figure 8.

Solving the optimization problem. Since the set of transitions in S is in general rather small due 1) the graph being very sparse and 2) the fact that we compute only a subset of transition around the gap, the computational benefit from using a sophisticated optimization algorithm is negligible. Hence, we simply compute exhaustively the value of the objective function for each of pairs T_1, T_2 and select the optimal pair.

E. Final reconstruction step

Up to now, the algorithm has only operated with a time resolution of the hop size $\tilde{a} = a\lceil \xi_s/\xi_{s,\max}\rceil$, where $a, \xi_s, \xi_{s,\max}$ are the parameters used in the computation of local audio features. It is quite possible that synthesizing directly using the selected transitions will produce timing mismatches of up to $\pm\tilde{a}$ samples. In order to reduce this phenomenon, we add a fine-tuning step before the reconstruction. As proposed in [8], we maximize the correlation between the regions around the proposed transition points. However, since the length of the resulting signal is allowed to change, we do it separately for each transition. More explicitly, let $t_{1,1}, \tilde{t}_{1,2}$ and $\tilde{t}_{2,1}, t_{2,2}$ be the proposed transition points around the gap and the proposed replacement segment respectively. Furthermore, for $j = 1, 2$,

$$s_{1,j}[l] = \begin{cases} s[t_{1,j} + l] & \text{if } |l| \le \tilde{w}_l, \\ 0 & \text{else} \end{cases}$$

and

$$s_{2,j}[l] = \begin{cases} s[t_{2,j} + l] & \text{if } |l| \le \tilde{w}_l + \tilde{a}, \\ 0 & \text{else,} \end{cases}$$

where $\tilde{w}_l = w_l\lceil \xi_s/\xi_{s,\max}\rceil$ and w_l is the size in samples of the (essential) support of the window $g \in \mathbb{R}^L$ used to compute the local audio features. Then

$$\epsilon_1 = \operatorname*{arg\,max}_{n\in]-a,a[} \langle s_{1,1}[\cdot - n], s_{2,1}\rangle, \quad \epsilon_2 = \operatorname*{arg\,max}_{n\in]-a,a[} \langle s_{1,2}[\cdot - n], s_{2,2}\rangle$$

and we obtain the final transition points for the replacement segment as

$$t_{2,1} = \tilde{t}_{2,1} + \epsilon_1, \quad t_{1,2} = \tilde{t}_{1,2} - \epsilon_2.$$

To obtain the reconstruction, we compute 3 STFTs, recombine the time frames appropriately and synthesize a signal. We define

$$\begin{aligned} C^{(1)}_{m,n} &= V_{\tilde{g}}s[n\tilde{a}, mL/\tilde{M}], \\ C^{(2)}_{m,n} &= V_{\tilde{g}}s[n\tilde{a} + \epsilon_1, mL/\tilde{M}], \\ C^{(3)}_{m,n} &= V_{\tilde{g}}s[n\tilde{a} - \epsilon_2, mL/\tilde{M}]. \end{aligned}$$

Note that $s \in \mathbb{R}^L$ now refers to the original signal before down-sampling. Hence we use the hop size $\tilde{a}$, $\tilde{M} = \tilde{w}_l$ and a new window $\tilde{g}$ with (essential) support of length $\tilde{w}_l$. The reconstruction is by applying the inverse STFT to

$$\begin{aligned} C^{\text{rec}} = \Big(\; & C^{(1)}_{\cdot,1}, \ldots, C^{(1)}_{\cdot,t_{1,1}/\tilde{a}-1}, \\ & C^{(2)}_{\cdot,t_{2,1}/\tilde{a}}, \ldots, C^{(2)}_{\cdot,t_{2,2}/\tilde{a}-1}, \\ & C^{(3)}_{\cdot,t_{1,2}/\tilde{a}}, \ldots, C^{(3)}_{\cdot,L/\tilde{a}-1} \Big). \end{aligned}$$

In fact, we only compute small segments of each $C^{(j)}$, $j = 1, 2, 3$, and insert the reconstruction at the correct part of the signal. Constructing the reconstruction in a time-frequency domain representation might seem overly complicated at first, but it conveniently introduces a cross-fading effect in the transition regions. Note also that simply measuring the correlation might not yield the optimal fine-tuning result. To see that, simply consider two signals with identical spectral content, but misaligned phase. In that case, only choosing the correct phase shift in addition to the proper time-alignment will produce optimal results. The implementation of such a scheme is currently in progress and more easily achieved in the time-frequency domain than in a pure time domain implementation.

IV. Algorithm evaluation

In this section we provide a first evaluation of our method's computational and qualitative performance. First, we evaluate the average runtime of the method on a modern personal computer (2.4 GHz Intel Core i7, 16 GB RAM). In order to show the reliability of the algorithm in a setting where the missing segment is also contained in the reliable data, we provide a synthetic *sanity check* in Section IV-B. Since this is an artificial assumption not usually exactly satisfied, and we do not even attempt to recover the same data, but a *perceptually pleasing* synthesis, a procedure for objective evaluation of the method's perceptual performance is not easily devised. In order to provide some indication of perceptual performance in more general situations anyway, we performed some preliminary perceptual tests, see Section IV-C, with promising results. Moreover, a MATLAB implementation of our algorithm, based on LTFAT [18] for feature extraction, and on the GSPBox [19] for graph creation is available for non-commercial use[5], alongside a browser-based demonstration available at https://lts2.epfl.ch/web-audio-inpainting/. For the sake of completeness, Table V at the end of the manuscript provides a list of the default parameters of the algorithm that have been used for all the presented experiments.

A. Computational complexity

The feature computation, graph creation and the selection of the optimal transition scale linearly with the length of the provided reliable data, in terms of both storage and time complexity. In all our experiments, the reliable data was given by a full song, without the corrupted segment. If multiple corruptions are to be removed using the same set of reliable data, the algorithm benefits from the fact that features only need to be computed once. Since the feature computation is the bottleneck of the method (this can be seen in Table IV-A), this leads to significant performance boosts. The Table IV-A shows mean computation time per minute (i.e. 44100 samples) after 80 runs of the algorithm on a set of 16 audio signals of various content, as well as the corresponding standard deviation. On average the algorithm requires 3.2s computation time per minute of single channel audio, sampled at 44.1 kHz.

Step	**Time (s)**	**Standard deviation (s)**
Features extraction	2.65	0.19
Graph construction	0.43	0.07
Transition selection	0.02	0.01
Signal reconstruction	0.05	0.02
Total	3.20	0.23

Table I
Average execution time per minute of music of the algorithm for a database of 16 songs.

B. Synthetic experiment

As a baseline check for the reliability of the algorithm, we address the question whether the algorithm consistently recovers the correct signal, whenever an exact, reliable copy of the corrupted segment is present in the set of reliable data. For this purpose, we used the same 16 signals as in the previous experiment and created new signals by just repeating the original content twice. Then, the signal was artificially corrupted by substituting a random segment of 2 seconds length with silence, before applying the method to that corruption. Per file, the experiment was repeated 5 times, leading once more to a set of 80 examples. In all cases, the ℓ^2-norm difference between the original signal and the reconstruction was in the range of numerical numerical precision, i.e. the signal was perfectly restored.

C. Preliminary perceptual validation

The previous experiment is, of course, insufficient to assess the performance of our algorithm in more realistic situations, where the exact same waveform may not appear in the reliable content. In order to estimate the potential of the proposed algorithm in a real-world situation, we have performed preliminary psychoacoustic experiments, rating the artifacts after restoration.

a) Methods: The sound material consisted of 134 songs from the genres classic, rock, pop, jazz, and others. From each song, three 5 s excerpts were selected as stimuli, each of them from a random position in the song. In order to create inpainted stimuli, two seconds within a stimulus was set to zero and then, based on the full song information, the inpainting algorithm was used to inpaint that stimulus part. Since the inpainting method may change the duration of the signal, it was limited to output stimuli with a maximal duration of 25 s.

Two listeners participated in the experiment. The listener EM was an expert on the sound material. The listener EA was an expert on the proposed inpainting algorithm.

In each trial, the listener was presented with a stimulus via headphones in their everyday listening conditions. After the stimulus presentation, the listener was asked to rate the

[5] https://lts2.epfl.ch/rrp/audio-inpainting/

stimulus by using one of the following categories: 1) inaudible artifacts or original stimulus; 2) audible but minor artifacts; 3) audible artifacts, not acceptable for listening; 4) strong artifacts, fail of the algorithm. Each stimulus was presented just once, yielding in 402 ratings per listener in total.

b) Results: The listener EM rated 38% of all stimuli as inaudible (category 1) and 65% of all stimuli as inaudible or minor artifacts (accumulated categories 1 and 2). The listener EA rated 72% of all stimuli as inaudible (category 1) and 86% of all stimuli as inaudible or minor artifacts (accumulated categories 1 and 2).

c) Discussion: Even though a complete conclusion cannot be drawn from that preliminary experiment, the results are very encouraging. On average, in 55% of all stimuli, the inpainting was rated as inaudible. In 76% of all stimuli, the inpainting result was rated as acceptable for listening. Note that the experiment was missing a hidden reference, thus, the reliability of the ratings as inaudible cannot be estimated. On the other had, the two listeners were experts, thus, it can be expected that for average listeners, the number of inaudible artifacts and thus successful reconstructions might have been even higher.

On average, 24% of the inpainted stimuli were rated as not acceptable for listening. The investigation of reasons for such ratings will require a more solid evaluation including hidden references and anchors. Further, since the proposed algorithm does not use any contextual or semantic information like language, audio material containing speech is clearly more challenging than instrumental music, thus, further in-depth perceptual evaluation will need to take this aspect into account.

V. Conclusion

We have introduced a method for restoration of audio signals in the presence of corruption/loss of data over an extended, connected period of time. Since, for complex audio signals, the length of the lost segment usually prohibits the inference of the correct data purely from the adjacent reliable data, our solution is based on the larger scale structure of the underlying audio signal. The reliable data is analyzed, detecting spectro-temporal similarities, resulting in a graph representation of the signal's temporal evolution that indicates strong similarities. Inpainting of the lost data is then achieved by determining two suitable transitions between the border regions around the corrupted signal segment and a region that is considered to be similar. In other words, the algorithm jumps from shortly before the *gap* to a similar section of the audio signal and, after some time, back to a position shortly after the gap, effectively exchanging the corrupted piece with a suitable substitute. Consequently, the algorithm is capable of efficiently exploiting naturally occurring redundancies in the reliable data. Although the internal similarity computations are not yet very closely related to human auditory perception, and incapable of detecting and comprehending semantic information, preliminary examination shows very promising results. Future work includes closing the gap between the internal similarity measures and human hearing by incorporating perceptually motivated similarity measures derived, possibly, from a perceptually-motivated representation [20] or a computational model of the auditory system [21]. Such a modification will greatly improve the reliability of the algorithm and its results. It seems worth noting, however, that even after considering an auditory model, reliable retrieval of strongly context-sensitive data such as speech and singing voice will require additional contextual information and might be better achieved by a generative approach [22], applied after separating voice and music in the signal [23].

Acknowledgment

We thank Pierre Vandergheynst for his support during this project. His ideas and suggestions have helped significantly to this contribution.

This work has been supported by the Swiss National Science Foundation research project *Towards Signal Processing on Graphs* (grant number: 2000_21/154350/1) and by the Austrian Science Fund (FWF) START-project FLAME (*Frames and Linear Operators for Acoustical Modeling and Parameter Estimation*; Y 551-N13)

References

[1] A. Adler, V. Emiya, M. G. Jafari, M. Elad, R. Gribonval, and M. D. Plumbley, "Audio inpainting," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 3, pp. 922–932, 2012.

[2] W. Etter, "Restoration of a discrete-time signal segment by interpolation based on the left-sided and right-sided autoregressive parameters," *IEEE Transactions on Signal Processing*, vol. 44, no. 5, pp. 1124–1135, 1996.

[3] D. J. Goodman, G. B. Lockhart, O. J. Wasem, and W.-C. Wong, "Waveform substitution techniques for recovering missing speech segments in packet voice communications," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 34, no. 6, pp. 1440–1448, 1986.

[4] K. Siedenburg, M. Dörfler, and M. Kowalski, "Audio inpainting with social sparsity," *SPARS (Signal Processing with Adaptive Sparse Structured Representations)*, 2013.

[5] A. Adler, V. Emiya, M. G. Jafari, M. Elad, R. Gribonval, and M. D. Plumbley, "A constrained matching pursuit approach to audio declipping," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 329–332.

[6] M. Lagrange, S. Marchand, and J.-b. Rault, "Long interpolation of audio signals using linear prediction in sinusoidal modeling," *J. Audio Eng. Soc*, vol. 53, no. 10, pp. 891–905, 2005. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=13390

[7] A. Lukin and J. Todd, "Parametric interpolation of gaps in audio signals," in *Audio Engineering Society Convention 125*, Oct 2008. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=14664

[8] Y. Bahat, Y. Y. Schechner, and M. Elad, "Self-content-based audio inpainting," *Signal Processing*, vol. 111, pp. 61–72, 2015.

[9] J. B. Allen and L. R. Rabiner, "A unified approach to short-time fourier analysis and synthesis," *Proceedings of the IEEE*, vol. 65, no. 11, pp. 1558–1564, 1977.

[10] K. Gröchening, *Foundations of time-frequency analysis*, ser. Applied and numerical harmonic analysis. Boston, MA, United States: Birkhaüser, 2001.

[11] C. Perkins, O. Hodson, and V. Hardman, "A survey of packet loss recovery techniques for streaming audio," *Network, IEEE*, vol. 12, no. 5, pp. 40–48, 1998.

[12] T. Jehan, "Creating music by listening," Ph.D. dissertation, Massachusetts Institute of Technology, 2005.

[13] ——, "Event-synchronous music analysis/synthesis," in *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx-04)*, 2004, pp. 361–366.

[14] S. Imai, "Cepstral analysis synthesis on the mel frequency scale," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'83.*, vol. 8. IEEE, 1983, pp. 93–96.

[15] F. Auger and P. Flandrin, "Improving the readability of time-frequency and time-scale representations by the reassignment method," *Signal Processing, IEEE Transactions on*, vol. 43, no. 5, pp. 1068–1089, May 1995.

Quantity	Variable used	Default value	Unit
Audio features			
Maximum sampling frequency	$\xi_{s,\text{max}}$	$12'000$	Hz
Size of the patch	a	128	samples
Number of frequencies	M	1024	-
Length of the window	L_w	M	samples
Type of window window	-	’itersine’	-
Dynamic range	p	50	dB
Trade-off between the amplitude and phase	λ	$2/3$	-
Graph			
Initial number of neighbours	k	40	-
Kernel length	l_k	40	-
Hard threshold for the weight matrix	t_w	2	-
Optimization			
Regularization parameter 1	γ_1	1	-
Regularization parameter 2	γ_2	100	-

Table II
DEFAULT PARAMETERS OF THE ALGORITHM

[16] N. Holighaus, Z. Pruša, and P. L. Søndergaard, “Reassignment and synchrosqueezing for general time-frequency filter banks, subsampling and processing,” *Signal Process.*, vol. 125, no. C, pp. 1–8, Aug. 2016. [Online]. Available: http://dx.doi.org/10.1016/j.sigpro.2016.01.007

[17] M. Muja and D. G. Lowe, “Scalable nearest neighbor algorithms for high dimensional data,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 11, pp. 2227–2240, 2014.

[18] Z. Průša, P. L. Søndergaard, N. Holighaus, C. Wiesmeyr, and P. Balazs, “The Large Time-Frequency Analysis Toolbox 2.0,” in *Sound, Music, and Motion*, ser. Lecture Notes in Computer Science, M. Aramaki, O. Derrien, R. Kronland-Martinet, and S. Ystad, Eds. Springer International Publishing, 2014, pp. 419–442. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-12976-1_25

[19] N. Perraudin, J. Paratte, D. Shuman, V. Kalofolias, P. Vandergheynst, and D. K. Hammond, “GSPBOX: A toolbox for signal processing on graphs,” *ArXiv e-prints*, Aug. 2014.

[20] T. Necciari, N. Holighaus, P. Balázs, and Z. Prusa, “A perceptually motivated filter bank with perfect reconstruction for audio signal processing,” *submitted, preprint available*, vol. abs/1601.06652, 2016. [Online]. Available: http://arxiv.org/abs/1601.06652

[21] T. Irino and R. D. Patterson, “A dynamic compressive gammachirp auditory filterbank,” *IEEE Audio, Speech, Language Process.*, vol. 14, no. 6, pp. 2222–2232, November 2006.

[22] K. Saino, H. Zen, Y. Nankaku, A. Lee, and K. Tokuda, “An hmm-based singing voice synthesis system.” in *INTERSPEECH*, 2006.

[23] Y. Li and D. Wang, “Separation of singing voice from music accompaniment for monaural recordings,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 4, pp. 1475–1487, 2007.