Proceedings of the 10th IASTED International Conference
SIGNAL AND IMAGE PROCESSING (SIP 2008)
August 18-20, 2008 Kailua-Kona, HI, USA

ENHANCING THE QUALITY OF AUDIO TRANSFORMATIONS USING THE MULTI-SCALE SHORT-TIME FOURIER TRANSFORM

Nicolas Juillerat
Pervasive and AI Research Group,
University of Fribourg, Switzerland.

Stefan Müller Arisona
Media Arts and Technology,
University of California, Santa Barbara.

Simon Schubiger-Banz
Computer Systems Institute,
ETH Zürich, Switzerland.

ABSTRACT
This paper presents a new adaptive tiling technique of the time-frequency plane that is suitable for a wide range of audio transformations. The proposed algorithm separates components of the audio signal into different categories according to their degree of transience. Each category is then processed with an adequate time-frequency resolution: higher time resolution is used for transient components and higher frequency resolution for steady sounds. The algorithm allows the audio signal to be modified and synthesized back with minimal interferences between the different components. An implementation of the proposed approach is presented and compared with other related approaches. The signal representation used by the presented algorithm is similar to that of a multi-channel short-time Fourier transform. Therefore, existing audio transformations based on the short-time Fourier transform can be adapted to the proposed approach with minimal modification, and automatically benefit from significantly improved quality: transient smearing artifacts for instance are mitigated without sacrificing quality on steady sounds. This includes a wide range of audio transformations such as pitch shifting, time stretching, chorusing, harmonizing, noise reduction, whisperization and various other audio effects.

KEY WORDS
Adaptive Tiling, Audio Effect, Audio Transformation, STFT, Time-frequency, Transient.

1. Introduction

The Short Time Fourier Transform (STFT) is a widely used tool for the implementation of various audio transformations. As any time-frequency analysis scheme, the STFT is subject to the uncertainty principle: no sound can be analyzed with both optimal time and optimal frequency resolutions. Time resolution can only be increased at the expense of frequency resolution and vice versa.

Another limitation of the STFT is that its time-frequency resolution is constant over both the time and frequency axes. All components of an audio signal are therefore analyzed with the same resolution, which is determined by the DFT size used.

As a consequence, for complex transformations, it is not always possible to find a satisfactory DFT size that works for audio signals with mixed content: a good time resolution (using a short DFT size) might work well for transient sounds (such as drums), while a good frequency resolution (using a large DFT size) might work well for steady sounds; but no choice gives good results on signals with both transient and steady sounds. Even with an optimal compromise, complex STFT-based audio transformations such a pitch shifting and time stretching are known to smear the transients or to introduce inaccuracies in steady sounds.

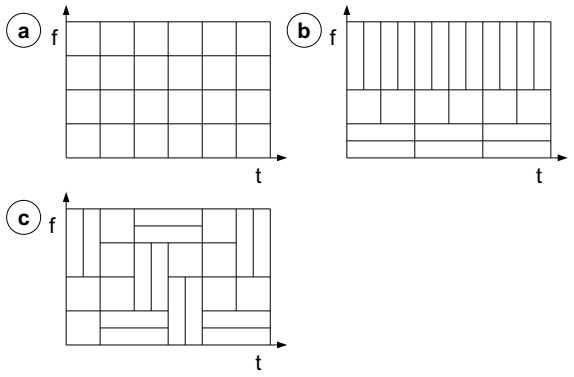

Figure 1. Different tilings of the time-frequency plane. (a) Rectangular, (b) multiresolution, (c) arbitrary

The STFT actually performs a fixed and rectangular tiling of the time-frequency plane, as shown in Figure 1a. In this figure, rectangles correspond to *tiles*, which are portions of the signal localized in both time and frequency. The uncertainty principle implies that the size of a tile cannot be arbitrarily reduced, although its shape can vary. Note that a rectangle represents the location in which a tile contains most of its energy. In practice, tiles are overlapping.

Several alternatives have been proposed to overcome the limitations of the STFT: wavelets [1, 2], wavelet packets [3, 4, 5], multi-resolution STFTs [6, 7], signal models [8, 9, 10, 11] and multi-scale Gabor analyses [12, 13]. Unfortunately, these alternatives make audio transformations harder to implement and introduce their own artifacts (see Section 2).

This paper presents the Multi-Scale Short-Time Fourier Transform (MS-STFT), a new algorithm that allows an adaptive tiling of the time-frequency plane to be performed, as illustrated by Figure 1c. The proposed algorithm performs a tiling that dynamically changes over

time according to the characteristics of the audio signal: transients for instance are automatically detected and analyzed with more time resolution than steady sounds. This is achieved by dynamically measuring the *transience* (or "transientness") of the signal in both time and frequency.

The proposed algorithm is not restricted to the *analysis* of audio signals. It also allows complex audio *transformations* to be performed and the result to be *synthesized* back with minimal interferences between components that have different time-frequency characteristics.

The MS-STFT is based on the STFT and produces a similar data structure. As a consequence most existing audio transformations based on the STFT can be adapted with almost no modification to the MS-STFT, and automatically benefit from improved quality. This includes a wide range of audio transformations such as pitch shifting, time stretching, chorusing, harmonizing, noise reduction, whisperization, metallization, etc.

The remainder of this paper is structured as follows: Section 2 gives an overview of related approaches. Section 3 reviews the STFT. Section 4 describes the MS-STFT and discusses how it was implemented. Results and comparisons with other approaches are given in Section 5.

2. Related Work

2.1 Multi-resolution

Multi-resolution approaches analyze higher frequencies with better time resolution, and lower frequencies with better frequency resolution. A typical multi-resolution tiling of the time-frequency plane is shown in Figure 1b. This scheme is motivated by the fact that most of the high frequency energy of a signal correspond to transients. Multi-resolution tilings are also known to be closer to the characteristics of the human auditory perception. Common implementations are based on wavelets [1, 2], or on multiple parallel STFTs [6, 7, 14].

Like for the STFT, a multi-resolution tiling does not change over time, and hence does not adapt itself to the signal characteristics. As a consequence, it can produce poor results in the presence of low frequency transients or high frequency steady sounds [8].

2.2 Wavelet Packets

Wavelet packets are an extension of wavelets that allow an arbitrary tiling of the time-frequency plane to be performed, as shown in Figure 1c. Implementations in which the tiling is changing over time according to the signal characteristics have been proposed [3]. With such an implementation, transient sounds such as drums are automatically analyzed with high time resolution while steady sounds are automatically analyzed with high frequency resolution. Wavelet packets have been used successfully for compression, noise reduction and coding [3, 4, 5].

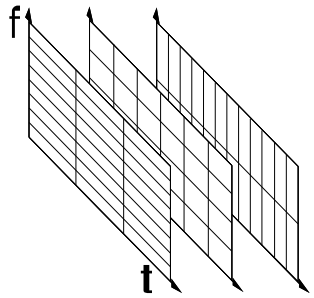

Figure 2. Rectangular tilings of the time-frequency plane with different resolutions

The implementation of a non-trivial audio transformation with wavelets or wavelet packets can produce complex interferences between the tiles. As they have heterogeneous time-frequency resolutions, properly coping with these interferences can be difficult in practice [8]. Furthermore, various audio transformations such as pitch shifting or time stretching require sound components to be "moved" from one location of the time-frequency plane to another. As the tiles may have different resolutions in the source and target locations, such a "move" involves a complex conversion from one resolution to another.

2.3 Multi-Scale Gabor Analysis

Multi-scale Gabor analysis [12] is a special case of the matching pursuit algorithm [15]. This approach uses a large dictionary of atoms consisting of windowed sinusoids with different locations in time and frequency, and with different time-frequency resolutions (also known as *scales*). The atoms of the dictionary fully cover the time-frequency plane multiple times with different time-frequency resolutions, as shown in Figure 2.

An iterative optimization algorithm attempts to select the smallest subset of atoms that most accurately approximates the audio signal. Atoms with high time resolution are automatically selected to approximate transients, and atoms with high frequency resolution to approximate steady sounds.

The limitation of this technique lies in the fact that the audio signal can only be *approximated*, and better approximation requires more iterations. Hence, this makes it well-suited for compression, but less adequate for high quality audio transformations. Nevertheless, an implementation of time stretching has been recently proposed [16]. However it requires an explicit processing of the residual signal in order to achieve high quality results.

2.4 Signal Models

The idea of these approaches is to decompose the signal into several components, such as sinusoids, transients and

noise, and to process each component separately [8, 9, 10, 11]. Sinusoids are processed with high frequency resolution while high time resolution is used for the transients.

The sines+transient+noise (STN) decomposition is considered one of the best approaches for the implementation of high quality pitch shifting and time stretching transformations. However, transients and noise require specific processing and there is no cue on how this can be adapted to other audio transformations.

STN decomposition separates transients from the rest of the signal as a part of the process. A limitation of this approach is that it does not cope well with sounds that are half-way between the two categories. In particular, transients that are "hidden" in a complex polyphonic music are usually not detected as such, and might be smeared after the transformation.

3. Background

3.1 The STFT

The STFT $X[s,k]$ of a discrete signal $x[t]$ is typically computed in two steps. In a first step the signal is divided into overlapped and windowed chunks (equation 1), and in a second step the DFT of the individual chunks is computed (equation 2):

$$x_s[n] = v[n]x[n+sR] \quad (1)$$

$$X[s,k] = \sum_{n=0}^{N-1} x_s[n]e^{-i\frac{2\pi kn}{N}} \quad (2)$$

where N is the DFT size, $v[n]$ is the analysis window, s is the STFT frame number and R is the analysis hop size. The s-th STFT *frame* X_s is defined as the vector formed by the N complex values $x[s,k]$ for $k = 0, 1, \ldots, N-1$. The short-hand notation X will be used to denote the serie of all STFT frames X_s.

An audio transformation can then be given as a function f of the STFT frames:

$$Y = f(X) \quad (3)$$

The inverse STFT used to synthesize the resulting signal from the transformed values $Y[s,k]$ is performed in two steps. First the inverse DFT of the STFT frames is computed (equation 4), and then the resulting chunks are combined with an overlap-add process (equation 5):

$$y_s[n] = \frac{1}{N}\sum_{k=0}^{N-1} Y[s,k]e^{i\frac{2\pi kn}{N}} \quad (4)$$

$$y[n] = \sum_{s=-\infty}^{\infty} w[n-sR']y_s[n-sR'] \quad (5)$$

where R' is the synthesis hop size and $w[n]$ is the synthesis window. Note that apart from transformations involving time stretching, the analysis hop size R and synthesis hop size R' are equal.

3.2 Stereo and Multi-Channel Processing

Multi-channel processing with the STFT plays an important role in understanding the MS-STFT, and is therefore summarized here.

When processing stereo and multi-channel audio signals, it is sometimes possible to process every channel independently. This works fine for various transformations such as filters, frequency shifting or noise reduction. However, for many audio transformations it is necessary to lock some parameters of the transformation between the channels. Failure to do so may result in phase differences between the channels, and to a loss of the stereo field [17].

To lock parameters between channels, a possible approach is to analyze these parameters not using the STFT frames of the current channel, but using the STFT frames M of a "common" signal that is built out of all channels. A simple choice is to just use the sum of the channels[1]:

$$M[s,k] = \sum_{c=0}^{C-1} X_c[s,k] \quad (6)$$

where $X_c[s,k]$ corresponds to STFT values of the channel c and C is the number of channels. A transformation that needs to lock some parameters between the channels can then be expressed as follows:

$$Y_c = f(X_c, M) \quad (7)$$

Given an audio transformation that does not lock any parameter, and is implemented according to the structure of an *adaptive digital audio effect* (as proposed in [19]), the necessary steps to lock some parameters of the transformation involve only minor modifications, namely:

- Adding the computation of the STFT values $M[s,k]$ using equation (6) as a part of the feature extraction process (before the transformation),
- Replacing references to $X_c[s,k]$ by $M[s,k]$ whenever the value directly or indirectly affects a parameter that needs to be locked.

In transformations based on the standard phase vocoder [20] for example, phases and frequencies are extracted from M while amplitudes are extracted from X_c. In the phase-locked vocoder using peak-picking [21, 22], amplitudes are extracted from M for the peak-picking stage, and from X_c for the actual transformation.

4. The Multi-Scale STFT

This section presents a new adaptive tiling technique, the multi-scale short-time Fourier transform (MS-STFT). It

[1] This is a simple scheme that works well in practice. In some cases it may be preferable to weight the result by $1/C$ or $1/\sqrt{C}$ [18]. More advanced approaches exist (for instance to better handle surround sounds encoded in a stereo stream) but are beyond the scope of this paper.

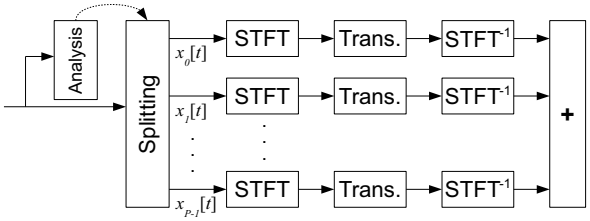

Figure 3. Structure of the MS-STFT

performs an arbitrary tiling (as illustrated by Figure 1c) that dynamically adapts itself according to the signal characteristics. The signal representation is similar to that of a multi-channel STFT, which allows complex audio transformations to be applied and the result to be synthesized back.

The global structure of the MS-STFT is illustrated by Figure 3 and can be summarized as follows:

- Analyze the transience of the signal in time and frequency,
- Split the signal into different *layers*, where each layer contains components of the signal with a given *degree of transience*,
- Analyze each layer with a different time-frequency resolution, using a modified STFT,
- Transform the resulting STFT frames as in a multi-channel audio transformation,
- Synthesize the resulting signal using a modified inverse STFT.

Each step is detailed in the following subsections.

4.1 Analysis of the Signal Transience

This first two steps of the algorithm separate the input signal $x[t]$ into different layers defined as $x_p[t]$, with $p = 0, 1, \ldots, P-1$. Each layer holds components of the signal within a given degree of transience. The sharpest sounds such as transients are contained in $x_0[t]$ while the smoothest sounds such as steady sinusoids are contained in $x_{P-1}[t]$. Note that $\sum_{p=0}^{P-1} x_p[t] = x[t]$ must hold.

The proposed implementation uses a cascade of transient detection/extraction algorithms with different parameters, as illustrated in Figure 4. Transient detection will be discussed first, and the next section will cover transient extraction.

Transient detection is well-covered in existing literature and several implementations have been proposed [23, 24, 25]. Almost all of them make use of a threshold at some point of the process; hence any implementation can be chosen and used in cascade, with different parameters and thresholds.

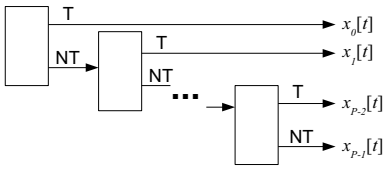

Figure 4. Cascade of transient extraction algorithms that each splits the input signal into transients (**T**) and non-transients (**NT**). With appropriate parameters, this results in multiple layers $x_p[t]$ corresponding to portions of the signal with different degrees of transience

In the proposed algorithm, the implementation of transient detection uses the STFT. It extends a previously proposed criterion whose idea is to compare the STFT magnitudes of a given frame with those of the previous frame as follows [24, 25]:

$$|X[s,k]| > \alpha |X[s-1,k]| \tag{8}$$

for a given $\alpha > 1$. The criterion (8) essentially detects sudden raises in energy. Note that this criterion is evaluated on every DFT bin. A transient, if any, is then built of all the DFT bins for which the criterion holds. This differs from some other approaches in which the entire STFT frame is either classified as transient or not [8, 9, 10].

The criterion (8) can be extended by replacing $|X[s-1,k]|$ by its "smoothed past" $\Omega[s,k]$ defined as follows:

$$\Omega[s,k] = \min_{j=0}^{J-1}(\omega[s-j,k]) \tag{9}$$

$$\text{where } \omega[s,k] = \frac{1}{I}\sum_{i=1}^{I}(|X[s-i,k]|) \tag{10}$$

for given values of I and J. In equation (10), the past DFT bin magnitudes are averaged. In equation (9), the minimum is taken among the past averaged values. The new criterion for transient detection is then:

$$|X[s,k]| > \alpha \Omega[s,k] \tag{11}$$

The criterion (11) has some advantages over the criterion (8) in practice: The averaging operation (equation 10) smoothes the values and minimizes the effects of random fluctuations due to noise and leakage, without requiring the use of larger DFT sizes. Keeping the DFT size small is important because high time resolution is necessary to accurately localize the transients in time.

The minimizing operation (equation 9) reduces the effects of a transient with a large magnitude on the averaged values $\omega[s,k]$, and therefore facilitates the detection of consecutive transients. Furthermore, both averaging and minimizing allow for better detection of less sharp transients, whose attacks are not abrupt and may span more than a single STFT frame.

A second improvement can be done based on the fact that most transients span many consecutive DFT bins over

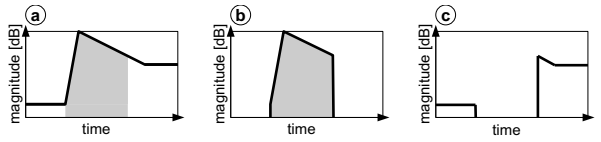

Figure 5. Traditional transient extraction on a single DFT bin: (a) initial signal, (b) transient, (c) non-transient

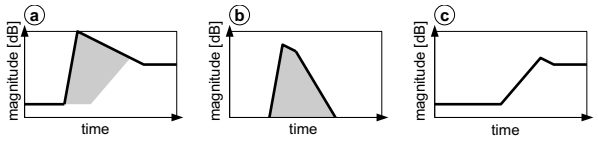

Figure 6. Proposed transient extraction on a single DFT bin: (a) initial signal, (b) transient, (c) non-transient

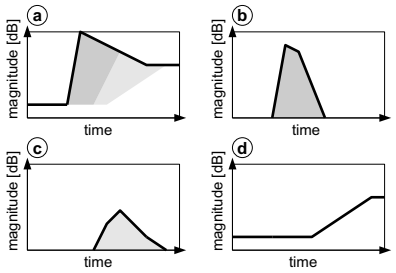

Figure 7. Illustration of the decomposition into three layers for a single DFT bin. (a) Initial signal, (b) first (most transient) layer, (c) second layer, (d) third (smoothest) layer

the frequency axis [24]. This fact can be used to avoid false positives for instance in the presence of a chirp that moves from one DFT bin to another. Assume that most transients span at least $D+1$ DFT bins for some $D \in \mathbb{N}$. The criterion (11) can be improved by keeping only the DFT bins for which at least D neighbors exist that also satisfy the criterion. This can be expressed as[2]:

$$\begin{array}{rl} \exists b \in [0, D] & \text{such that} \\ |X[s, k-b+d]| & > \alpha \Omega[s, k-b+d] \\ & \forall d \in [0, D] \end{array} \quad (12)$$

Listening tests have shown the criterion (12) to be significantly better than the criterion (11) in practice, especially for the detection of very short transients. In particular, it allows a smaller value of α to be chosen without introducing more false positives.

Different DFT sizes and different values of I, J, α and D are used for the different transient detections of the full decomposition process illustrated in Figure 4. Like with any transient detection scheme, the choice of these values involves some fine-tuning in practice. See Section 5.1 for possible choices.

4.2 Splitting the Signal Into Layers

Once the transients have been detected with the method described in the previous section, they have to be extracted. A simple scheme is to use criterion (12) to select the STFT values $X[s,k]$ corresponding to transient and non-transient parts of the signal. This is similar to previously proposed approaches [8, 9, 10] as illustrated in Figure 5 for a single DFT bin.

The proposed algorithm uses an improved approach that considers the evolution of the magnitudes of each DFT bin over time, and allows only a fraction of an STFT value to be extracted as a transient. First, two helper variables are defined:

$$\begin{array}{rcl} \tau[s,k] & = & \begin{cases} \min(\Omega[s,k], \beta\tau[s-1,k]) & \text{if (12)} \\ \max(\beta\tau[s-1,k], \epsilon), & \text{else} \end{cases} \\ \kappa[s,k] & = & \min(\tau[s,k], |X[s,k]|) \end{array}$$

with some $\beta > 1$. The constant ϵ is here to ensure that $\tau[s,k]$ is always greater than zero. A common value for ϵ is the threshold of hearing or quantization, such as -96dB. Note that in practice, $\tau[s,k]$ is defined to be zero for values of s before the beginning of the signal.

Then the transient part $X'[s,k]$ and the non-transient part $X''[s,k]$ are defined as:

$$\begin{array}{rcl} X'[s,k] & = & (1 - \frac{\kappa[s,k]}{|X[s,k]|}) X[s,k] \\ X''[s,k] & = & \frac{\kappa[s,k]}{|X[s,k]|} X[s,k] \end{array} \quad (13)$$

The idea of the process is illustrated in Figure 6 for the time evolution of the magnitude of a single DFT bin. When a transient is detected, only the portion of the signal that is above its "smoothed past" $\Omega[s,k]$ is extracted as a transient. In subsequent frames, the transient part is not stopped abruptly, but *faded out*. The fade out "speed" is determined by the β coefficient. A different value is chosen for the extraction of each transience layer. A faster fade-out for instance is used for layers corresponding to the most transient components.

The signal of Figure 6a may correspond to the sum of a soft steady note and a loud piano note. The improved definition (13) has several advantages over a straightforward application of the threshold criterion. The use of β ensures that the non-transient part fades in smoothly and is free from any abrupt change of magnitude. Furthermore, the definition (13) allows only the part of the signal that is above the level of the soft steady note to be captured. When applied on all DFT bins, this separates transients effectively even on polyphonic audio signals. This differs from existing transient extraction schemes in which no part of the signal in the time-frequency plane can belong to both the transient and non-transient parts [23, 24, 25].

[2] In Equation (12) an STFT value $X[s,k]$ is defined to be zero for k outside of the $[0, N-1]$ interval.

Figure 7 illustrates the result when a cascade of two transient extractions are used to get three transience layers. Note that layers corresponding to smoother components contain a proportionally smoother version of the attack. This differs from other proposed transient extraction schemes [8, 9, 10] illustrated in Figure 5, in which the non-transient part still contains an abrupt change of the magnitude. The latter is not desirable because lower time resolution will be used for the smoother layers. Thus any abrupt change of the signal level may result in smearing artifacts. With the proposed approach, smearing artifacts on the smoother layers have little impact as these layers are already smeared by definition.

4.3 Analyzing the Layers

The next step of the algorithm is to analyze each layer using the STFT. Obviously, higher time resolution must be used for more transient layers and higher frequency resolution for smoother layers. A possibility is to use different DFT sizes for that purpose. Unfortunately, this would create a heterogeneous representation of the signal, and the implementation of a transformation would be subject to the same limitations found in the existing techniques discussed in Section 2.

The proposed solution is to first choose a large DFT size N that is suitable for the smoothest layer. Then the idea is to use it for all layers, and to shrink the analysis window in order to increase the time resolution on more transient layers.

The amount of shrinking is obviously higher for more transient layers. Given an analysis window $v[t]$ with support on the $[0..N-1]$ interval, a shrunk version $v_p[t]$ for processing the signal $x_p[t]$ of the layer p can be given by:

$$v_p[t] = \sqrt{\xi_p} v[\lfloor \frac{N-1}{2} + \frac{1}{\xi_p}(t - \frac{N-1}{2}) + 0.5 \rfloor] \quad (14)$$

where $\xi_p \geq 1$ is the shrinking factor for the layer p. Note that equation (14) does not interpolate the window coefficients. While an interpolation could be used in practice, this has not shown to provide significant improvements.

Using a shrunk analysis windows is similar to performing a N-point DFT on a chunk of length N/ξ_p with *zero padding* [26]. In other words, shrinking the analysis window reduces the frequency resolution (and increases the time resolution) without changing the DFT size [27].

To ensure that the STFT frames have the same characteristics in all layers, they must also all use the same analysis hop size. The hop size must be chosen smaller than the smallest window after shrinking in order to allow reconstruction.

The initial window $v[t]$ and the shrunk windows $v_p[t]$ are all centered in the middle of the chunk: this ensures that the corresponding STFT frames all have the same time location. Note the $\sqrt{\xi_p}$ factor used in Equation (14) whose purpose is to preserve the energy of the analysis window.

Using this approach, all layers exhibit the *same* tiling of the time-frequency plane. This is a rectangular tiling similar to that of the standard STFT (see Figure 1a), except that the tiles are much smaller. The time-frequency resolution is not magically increased though, because larger *correlation* is introduced between the tiles. In the smoothest layers, there is a large correlation between consecutive tiles along the time axis, because the frames are heavily overlapped. In the most transient layers, there is large correlation between consecutive tiles along the frequency axis, due to the use of shrunk analysis windows.

4.4 Audio Transformation

The main advantage of the approach proposed in the previous section is the following: the STFT frames have the same size and time locations in all layers. Hence all the layers can just be processed *as a single multi-channel signal*, where the number of channels to process is given by the number of layers P.

In the general case where the input signal is already a multi-channel signal (such as a stereo signal) with C audio channels, the separation into layers is performed on each channel. Then the number of resulting channels to process is given by the product of P by C.

If an STFT-based implementation of an audio transformation already exists, and correctly locks the relevant parameters between the channels as discussed in Section 3.2, it can be used *as is* with the presented MS-STFT. Indeed, the locking of parameters between the audio channels will automatically lock them between the different layers, because they are "seen" as channels by the transformation algorithm. Locking ensures that no undesirable phase cancellations or other interferences occur between the different layers in the result.

If an implementation of an audio transformation exists for the STFT, but processes each channel independently, it might be necessary to modify it such that it locks some parameters between the layers. This typically involves a few modifications only, as detailed in Section 3.2. This concerns for example noise reduction techniques such as noise gating and spectral subtraction.

4.5 Synthesis

The simplest synthesis scheme would be to apply the inverse STFT on individual layers, and then to sum the results to get the final transformed signal. It is possible to do much better though.

Recall that shrunk analysis windows $v_p[t]$ with supports of lengths N/ξ_p possibly smaller than the DFT size N are used in the analysis. Computing the inverse DFT of the STFT frames yields chunks with the same support lengths in the absence of any transformation. If a transformation is applied though, the inverse DFT may yield chunks with larger support lengths, up to N. The reason of this "smearing" is that various audio transformations (in

particular those based on the phase vocoder [20, 21]) do not maintain the relative phases of the DFT bins. Another way of understanding the problem is that the use of a shrunk analysis window decreases the frequency resolution of the STFT frames, and hence introduces correlation between the DFT bins. A transformation may destroy this correlation and hence enlarge the underlying chunk.

This problem is undesirable because it would void the benefits of using different time-frequency resolutions in the presence of smearing artifacts in the transformation. Therefore the algorithm uses synthesis windows that are shrunk in the same way as the corresponding analysis windows (see equation 14). Indeed, the application of a shrunk synthesis window "crops" the part of the chunk that is smeared outside of the window support.

To preserve the energy contained in the chunk during this process, its samples must be multiplied by a correction factor E_N/E_s, where E_N is the energy of the whole chunk and E_s is the energy of the chunk within the window support.

This correction factor may get large (even infinite) in case of severe smearing. In practice though, it was observed that the smearing introduced by most audio transformations rarely requires a correction factor substantially larger than the shrinking factor ξ_p. Limiting the correction factor to ξ_p works well in practice. Also note that in the absence of any audio transformation, no smearing occurs, and the entire process exactly reconstructs the input signal.

5. Results

5.1 Implementation

PitchTech is an existing Java framework for STFT-based digital audio transformations that was presented in a previous paper [28]. This framework cleanly separates the computation of the STFT and its inverse from the actual audio transformations. In other words, audio transformations are functions of STFT frames (as in equations 3 and 7) and not of the input signal $x[t]$; they do not handle the analysis and synthesis processes. By sequencing an STFT, an audio transformation and an inverse STFT, a "full" audio transformation of the signal $x[t]$ is built. *PitchTech* implements several audio transformations with the STFT, and with correct support for stereo and multi-channel audio signals using the techniques discussed in Section 3.2. They all support user-defined DFT sizes and hop sizes.

The analysis part (including the separation of the signal into layers) and the synthesis part of the MS-STFT presented in this paper have both been implemented as extensions of the existing *PitchTech* framework.

Once this was done, existing STFT-based audio transformations were adapted to the MS-STFT with almost no modifications of the existing code. For noise reduction for instance, the only necessary modifications were those suggested in Section 3.2. For all audio transformations based on the phase vocoder [20] and its variants [21, 22], *no modification of the existing code was necessary,* as predicted in Section 4.4. This includes a wide range of audio transformations, namely pitch shifting, time stretching, chorusing, harmonizing, and various other special audio effects.

N	128	512
$R = R'$	32	128
I	20 (14.5ms)	38 (110.3ms)
J	24 (17.4ms)	38 (110.3ms)
α	5dB	1dB
D	9	4
β	0.7dB (965dB/s)	2dB (689dB/s)

Table 1. Parameter values used in the two transient detection and extraction processes

N	R	ξ_0	ξ_1	ξ_2
8192	128	16	4	1

Table 2. Parameter values used in the analysis and synthesis processes

Only a few audio transformations did not work properly with the MS-STFT, because they rely on specific (or unusual) DFT or hop sizes, such as robotization and one of the implementations of whisperization [18].

The current implementation of the MS-STFT is a prototype based on three layers only, corresponding to three degrees of transience. It uses two transient detection algorithms and three modified STFTs. The values used for the various constants mentioned in the equations of the previous sections are given by Tables 1 and 2, for a sample rate of 44.1kHz. Table 1 gives the values related to the transient detection processes used for the separation of the signal into transience layers. Mind that I, J and β are relative to STFT frames, D is in DFT bins, and N and R are in samples. Table 2 gives the values related to the modified STFTs used for the analysis and synthesis processes. Fine tuning these values is still a matter of future works.

Other future investigations include the testing and comparison of alternate transient detection schemes for the decomposition of the signal into transience layers.

5.2 Performance Considerations

The MS-STFT is obviously slower than the standard STFT. The separation of the signal into layers introduces a fixed cost, and multiplies the processing time by the number of layers. However, the major factor is the ratio between the analysis window sizes of the least and most transient layers. The least transient layer uses the largest window size, while the most transient one uses the smallest window size (see Section 4.3). Yet the window size of the least transient layer

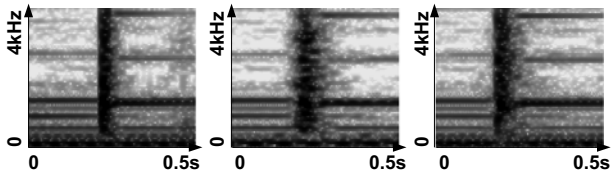

Figure 8. Spectrogram of a music excerpt featuring a short drum hit (left), time-stretched using the STFT (center) and using the MS-STFT (right)

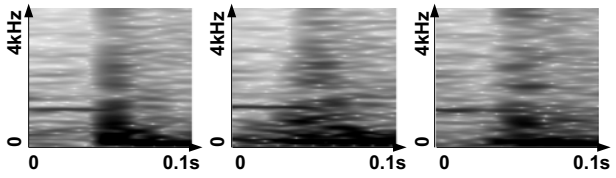

Figure 9. Spectrogram of a hidden transient (left), time-stretched using a multiresolution approach (center) and using the MS-STFT (right)

imposes a minimum DFT size, whereas the window size of the most transient one imposes a maximum hop size. Large DFT sizes and small hop sizes both imply slower processing.

An interesting consequence of this observation is that the required computing power increases with the maximum difference between the used time-frequency resolutions (between those of the least and most transient layers). With the implementation detailed in the previous section, noise gating and pitch shifting (using the fast phase-locked vocoder proposed in [21]) both run in real-time: they take about 90% of the CPU on a 2.4 GHz Pentium processor with the Sun Java virtual machine, version 6. Some other transformations are slower than real-time on the same architecture. Yet by reducing the number of layers, or the maximum difference between the time-frequency resolutions, faster versions can be implemented, at the expense of a lower quality. The implementation detailed in the previous section uses windows of sizes 512, 2048 and 8192. An alternate implementation using windows of sizes 1024, 2048 and 4096 has shown to run about 4 times faster at the expense of a lower quality, but still higher than the standard STFT.

At one extreme, a degenerated version with a single layer is just equivalent to, and as fast as a version using the standard STFT. At the other extreme, even more layers and larger differences in the chosen time-frequency resolutions could be used for non real-time applications in which quality is more important than speed.

5.3 Comparison With Other Approaches

Various audio transformations have been applied on several musical signals, with both the standard STFT and the MS-STFT. Audio excerpts are available on-line [29]!! Apart from audio transformations that are free (or almost free) from audible artifacts such as frequency shifting and simple graphic equalizers, the MS-STFT shows to constantly produce better results. In particular, the standard STFT exhibits transient smearing when large DFT sizes are used, and various frequency errors when small DFT sizes are used. When an optimal DFT size is chosen, both artifacts are usually present. These two problems are significantly reduced with the use of the MS-STFT, and are hardly audible in most cases.

Figure 8 shows the spectrograms of an audio signal time-stretched with the phase-locked vocoder using the standard STFT and the MS-STFT. The signal features a strong drum hit over soft noisy and steady sounds. In the version transformed with the standard STFT, the drum is smeared in time, and the amplitudes of other components are randomly attenuated (especially the background noise following the drum hit). These two problems are much less pronounced in the version transformed using the MS-STFT. Note that changing the DFT size in the standard STFT version can only reduce one of the two artifacts at the expense of the other.

Figure 9 shows the spectrograms of a transient time-stretched with a multiresolution approach (as discussed in [6]) and with the MS-STFT. The multiresolution approach smears the transient in the low frequencies. This artifact is much less pronounced with the MS-STFT. Interestingly, the transient is "hidden" by several other sounds. As a consequence, it is not captured by the most transient layer of the MS-STFT, but only by the second, intermediate layer. Some smearing is therefore introduced, but less than if it were captured by the least transient layer. The introduced smearing hence remains sufficiently low to be mostly masked by the other components of the signal. This differs from approaches based on a single separation between transients and non-transients. In such approaches, the transient could hardly be detected as such without capturing many false positives as well. But failure to detect it as a transient is likely to introduce more smearing than with the MS-STFT, as non-transients are handled with the highest frequency resolution in these approaches.

6. Conclusion

This paper has presented a new adaptive tiling algorithm based on the Short Time Fourier Transform (STFT). The proposed algorithm is able to dynamically detect the degree of transience of individual components of an audio signal and to automatically process them with adequate time-frequency resolutions: transient sounds such as drums are processed with high time resolution whereas steady sounds are processed with high frequency resolution.

Although the proposed algorithm performs an adaptive tiling of the time-frequency plane, it uses a homogeneous signal representation. With the help of techniques

coming from multi-channel audio processing, it hence allows the signal to be transformed and synthesized back with minimal interferences. Based on these ideas, an implementation was proposed and realized, which can be used in a way that is similar to the standard STFT.

Several existing audio transformations based on the STFT were adapted to the new approach with almost no modification. This includes a wide range of audio transformations such as pitch shifting, time stretching, chorusing, harmonizing, whisperization, metallization, noise gating, and other exotic audio effects. This is an improvement compared to previously proposed approaches such as multi-resolution schemes, wavelet packets or multi-Gabor analysis, with which only a restricted number of audio transformations have been successfully implemented yet.

The new approach was implemented and compared with the STFT on various audio signals with several audio transformations. It was shown to constantly provide significant improvements in terms of the quality of the result. In particular, the transient smearing artifact is mitigated, without sacrificing quality on steady sounds.

References

[1] A. Kumar, & R. Jain, Speech pitch shifting using complex continuous wavelet transform, *Proc. of the IEEE Annual India Conference*, 2006, 1 – 4.

[2] A.G. Sklar, *A Wavelet-based pitch-shifting method*, http://umsis.miami.edu/∼asklar/pitchshift.pdf, 2006.

[3] R. Bernardini, & J. Kovačević, Arbitrary tilings of the time-frequency plane using local bases, *IEEE Transactions on Signal processing, 47*(8), 1999, 2293 – 2304.

[4] M.M. Goodwin, *Adaptive signal models: theory, algorithms and audio applications*, PhD at the Massachusetts Institute of Technology, 1997.

[5] J. Horng, & R.A. Haddad, Variable time-frequency tiling using block transforms, *Proc. of the IEEE Digital Signal Processing workshop*, 1996, 25 – 28.

[6] D. Dorran, *Audio time-scale modification* , PhD Thesis at the Dublin Institute of Technology, 2005.

[7] J.B. Sanjaume, *Audio time-scale modification in the context of professional post-production*, PhD Thesis at the university Pompeu Fabra, Barcelona, 2002.

[8] S.N. Levine, *Audio representation for data compression and compressed domain processing*, PhD at the University of Stanford, 1998.

[9] S.N. Levine, & J.O. Smith, A sines+transients+noise audio representation for data compression and time/pitch scale modifications, *Proc. of the 105th Audio Engineering Society Convention*, San Francisco, 1998.

[10] T.S. Verma, & T.H.Y. Meng, Time scale modification using a sines+transients+noise signal model, *Proc. of the Digital Audio Effects Workshop*, Barcelona, 1998, 49 – 52.

[11] X. Serra, & J. Smith, Spectral modeling synthesis: a sound analysis/synthesis based on a deterministic plus stochastic decomposition, *Computer Music Journal, 14*(4), 1990, 12 – 24.

[12] F. Jaillet, & B. Torrésani, Time-frequency jigsaw puzzle: adaptive multiwindow and multilayered Gabor expansions, *International Journal of Wavelets, Multiresolution and Information Processing, 5*(2), 2007, 293 – 316.

[13] P.J. Wolfe, S.J. Godsill, & M. Dörfler, Multi-Gabor dictionaries for audio time-frequency analysis, *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2001, 43 – 46.

[14] S.N. Levine, T.S. Verma, & J.O. Smith, Multiresolution sinusoidal modeling for wideband audio with modifications, *IEEE International Conference on Acoustics, Speech and Signal Processing*, Seattle, WA, 1998, 3585 – 3588.

[15] S. Mallat, & Z. Zhang, Matching pursuit with time-frequency dictionaries, *IEEE Transactions on Signal Processing, 41*(12), 1993, 3397 – 3415.

[16] O. Derrien, Time-scaling of audio signals using multi-scale Gabor analysis, *Proc. of the 10th Int. Conference on Digital Audio Effects*, 2007, 1 – 6.

[17] E. Ravelli, M. Sandler, & J.P. Bello, Fast implementation for non-linear time-scaling of stereo signals, *Proc. of the 8th Int. Conference on Digital Audio Effects*, 2005, 182 – 185.

[18] U. Zölzer, *DAFX - Digital audio effects* (John Wiley & Sons, 2002).

[19] V. Verfaille, U. Zölzer, & D. Arflb, Adaptive digital audio effects (a-DAFx): a new class of sound transformations, *IEEE Transactions on Audio, Speech and Language Processing, 14*(5), 2006, 1817 – 1831.

[20] J.L. Flanagan, & R.M. Golden, Phase vocoder, *Bell Syst. Tech. Journal, 45*, 1996, 1493 – 1509.

[21] J. Laroche, & M. Dolson, New phase-vocoder techniques for real-time pitch-shifting, chorusing, harmonizing and other exotic audio modifications, *Journal of the Audio Engineering Society, 47*(11), 1999, 928 – 936.

[22] J. Laroche, & M. Dolson, Improved phase vocoder time-scale modification of audio, *IEEE Transactions on Speech and Audio Processing, 7*(3), 1999, 323 – 332.

[23] W.A. Sethares, *Rhythm and transforms* (Springer, 1st edition, 2007).

[24] L. Daudet, A review on techniques for the extraction of transients in musical signals, *Lecture Notes in Computer Science, 3902*, 2006, 219 – 232.

[25] J.P. Bello & al., A tutorial on onset detection in music signals, *IEEE Transactions on Speech and Audio Processing, 13*(5), 2005, 1817 – 1831.

[26] R.G. Lyons, *Understanding digital signal processing* (Englewood Cliffs, NJ: Prentice-Hall, 2004).

[27] K. Dressler, Sinusoidal extraction using an efficient implementation of a multi-resolution FFT, *Proc. of the 9th Int. Conference on Digital Audio Effects*, 2006, 247 – 252.

[28] N. Juillerat, S. Müller Arisona, & S. Schubiger-Banz, Real-time, low latency audio processing in Java, *Proc. of the Int. Computer Music Conference, vol. II*, Copenhagen, Denmark, 2007, 99 – 102.

[29] http://www.pitchtech.ch/Confs/SIP2008/index.html