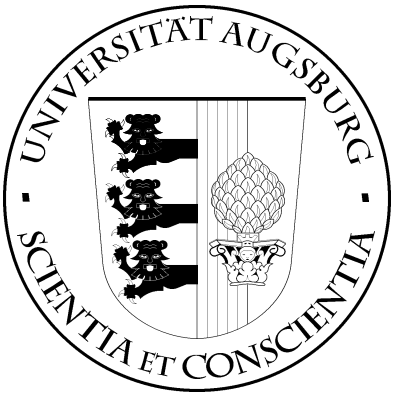
Universität Augsburg

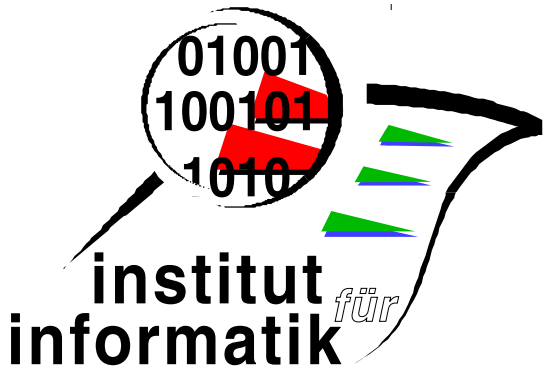


Audio Brush: Editing Audio in the Spectrogram

C. G. v. d. Boogaart, R. Lienhart

Report 2005-10

Juni 2005
revised Oktober 2005



Institut für Informatik
D-86135 Augsburg

Audio Brush: Editing Audio in the Spectrogram

C. G. v. d. Boogaart, R. Lienhart
Multimedia Computing Lab
University of Augsburg
86159 Augsburg, Germany
{boogaart,lienhart}@informatik.uni-augsburg.de

ABSTRACT

A tool for editing audio signals in the spectrogram is presented. It allows manipulating the spectrogram of a signal at any chosen time-frequency resolution directly and to reconstruct the edited signal in HiFi quality – a capability that is usually not possible with the Fourier or wavelet transformation. Image processing and computer vision methods are applied to the spectrogram in order to identify, separate, eliminate and/or modify audio objects visually. As spectrograms give descriptive information about the sound, this tool allows editing audio in a "what you see is what you hear" style. This is enabled by a thorough investigation and exploitation of Gabor analysis and synthesis. We further propose to use a kind of zooming, as in visual painting tools, which results in a change of time and frequency resolution, and can be adapted for the task at hand. Results on applying this tool to erasing audio objects such as whistles, music, clapping and alike in audio tracks are presented. Hence audio objects are automatically identified as visual objects in the spectrogram and eliminated therein. The cleaned signal is then reconstructed from the spectrogram in HiFi quality.

1. INTRODUCTION

Hearing, analyzing and evaluating sounds is possible for everyone. The reference-sensor for audio, the human ear, is of amazingly high quality. In contrast editing and synthesizing audio is an indirect and non-intuitive task, which needs more expertise. This is normally performed by experts using specialized tools for audio-effects such as a lowpass-filter or a reverb. This situation is depicted in figure 1: A user can edit a given sound by sending it through an audio-effect (1). The input (2) and the output (3) are evaluated acoustically and sometimes also with a spectrogram (4,5). The audio-effects can only be controlled via some dedicated parameters (6) and therefore allow editing on a very abstract and crude level. To generate best results with this technique it is state of the art to record every sound separately on a different track in clean studio conditions.

The effects can now be applied to each channel separately. More direct audio editing is desirable, but it is not possible up to today. This limitation to indirect and non-intuitive editing is due to the lack of a human reference audio-actuator equivalently flexible as the reference-sensor human ear.

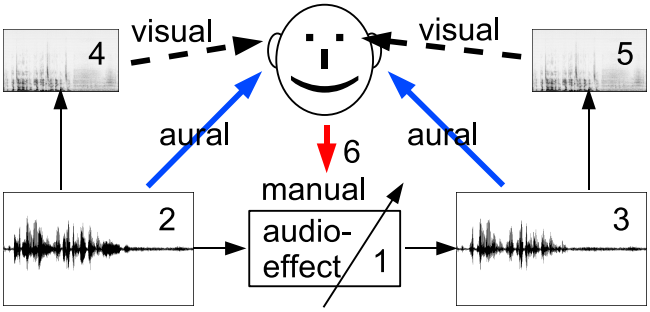


Figure 1: Classical situation in audio editing: A sound is sent through an audio-effect (1). The input (2) and the output (3) are evaluated acoustically and sometimes visually (4,5). The audio-effects are controlled via some dedicated parameters (6).

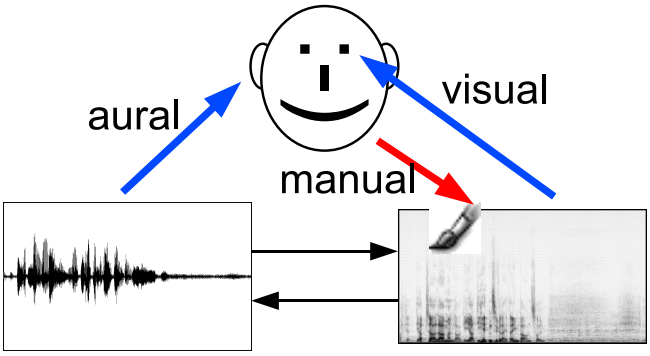


Figure 2: Editing with Audio Brush: The spectrogram of a sound is edited directly. The result can be evaluated either visually or acoustically.

The goal of Audio Brush is to lower these limitations by providing a means to directly and visually edit audio spectrograms, out of which high quality audio can be reproduced. Figure 2 shows the new approach: A user can edit

the spectrogram of a sound directly. The result can be evaluated either visually or acoustically resulting in a shorter closed loop for editing and evaluation. This has several advantages:

1. A spectrogram is a very good representation of an audio-signal. Often speech-experts are able to read text out of speech-spectrograms. In our approach, the spectrogram is used as representation of both, the original and the recreated audio-signal, which both can be represented visually and acoustically. It therefore narrows the gap between hearing and editing audio.

2. Audio is transient. It is emitted by a source through a dynamic process, travels through air and is received by the human ear. It cannot be held for investigation at a given time moment and frequency band. This limitation is overcome by representing the audio signal as a spectrogram. The spectrogram can be studied in detail and edited appropriately before inverse transforming it back into the dynamic audio domain.

Figure 3 gives an overview over the several stages of the Audio Brush tool-chain. A time signal (1) is transformed (2) into a manifold of time-frequency representations (3). One representation is chosen (4) and edited (5). By inverse transformation (6) an edited time signal (7) is reproduced. By the appropriate choice of one of the manifold time-frequency representations, which refer to higher time or higher frequency resolution, it is possible to edit with high accuracy in time and frequency. If necessary, the process is repeated (8).

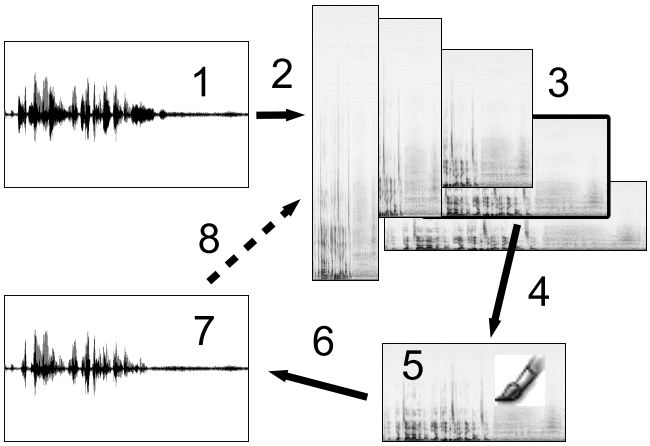


Figure 3: Overview of the Audio Brush tool-chain: a time signal (1) is transformed (2) into a manifold of time-frequency representations (3). One representation is chosen (4) and edited (5). By inverse transformation (6) an edited time signal (7) is recreated. If necessary, the process is repeated (8).

1.1 Related work

Time-frequency transformations are a well-known and intensively used tool in automatic processing of audio signals. Standard transformations are: wavelets ([4], [18]), DFT and FFT ([15]) and Wigner-Ville-Distribution ([16]). In audio processing they are mainly used for either calculation of features or for compression and filtering. In the first case, the features are used for recognizing speech ([12]), speakers ([17]) or music pieces ([9]). There is no way back from the features to the audio signal. An inverse transformation is not performed, but the features are used to derive higher semantics from the audio signal. In the second case, e.g. lossy speech compression ([18]) or denoising ([5]), an inverse transformation is performed and the signal is either intentionally or unintentionally edited in the short term frequency domain. However the editing is not based on any visualization and thus no visual manipulation concepts can be applied.

Multiresolution Gabor analysis has already been applied for audio analysis (see [21]). The focus is on using overcomplete gabor dictionaries, which reveal transient and tonal parts of the signal separately.

1.2 Outline

The paper is organized as follows: In Section 2 we discuss the design and general implementation issues of an appropriate forward and inverse transformation based on the Gabor transformation. We propose using the Gaussian as window function due to its unique localization property, discuss the choice of the lattice constants and oversampling factor and give remarks on implementation issues like discretization and truncation. In Section 3 we discuss the choice of the appropriate time-frequency resolution out of the manifold based on physical facts. In Section 4 we discuss and give examples for detection of known audio objects and removal of detected audio objects from an audio track. Section 5 concludes with a short summary of the main aspects.

2. INVERTIBLE TIME-FREQUENCY TRANSFORMATIONS

There are several common time-frequency transformations, which are invertible and have applications in the field of audio, such as wavelets, DFT, FFT and Wigner-Ville-Distribution. In contrast to the Fourier transformation[1] they all have the same idea in common: They transform the temporal signal into a joint time-frequency domain by windowing in the time domain. Nevertheless, they have some disadvantages for the purpose of editable spectrograms, which led us to the use of the Gabor transformation. The Gabor transformation described in this chapter is invertible and creates perfectly localized transformation results because of the following properties:

- The window-functions are localized in time and frequency.
- The optimal time-frequency resolution according to the Heisenberg uncertainty principle is reached.
- The transformation result is interference free, i.e. the influence of the coefficients decays strictly in time and frequency direction.

The ideas of the Gabor transformation (Gabor regression, Gabor expansion) trace back to Dennis Gabor [8]. Since its original formulation the theory has been developed much further, revealing serious problems in theory and possible

[1]The Fourier transformation computes the spectrum of a time signal, hiding all information of its temporal evolution in the phases, i.e. it transforms a time signal into a pure frequency signal (see [14]).

solutions to deal with them. The major results can be found in [6] and [7].

2.1 Fundamentals of the Gabor transformation

As the Gabor transformation is its discretized version, we start our discussion with the *windowed* or *short time Fourier transformation* (STFT). It was developed to overcome the lack of time localization of the Fourier transformation. The STFT is defined as follows: A time function $x(t)$ is split in the time-frequency domain into $X(t, f)$ by the use of a windowing function $w(t)$:

$$X(t,f) = \int_{-\infty}^{+\infty} x(\tau) w^*(\tau - t) e^{-j2\pi f\tau}\, d\tau. \quad (1)$$

This is the inner product of the temporal signal $x(t)$ with a modulated (by $e^{-j2\pi ft}$) and time shifted, conjugate complex window function $w(t)$. The inner product measures the similarity of the signal to the so called prototype function $w^*(\tau - t)e^{-j2\pi f\tau}$ ([18]). To get the local properties of $x(t)$ the window $w(t)$ has to be chosen appropriately to be localized in time and frequency. An inverse transformation reconstructs the signal and is given by the formula ([1]):

$$x(t) \int_{-\infty}^{+\infty} |w(t)|^2\, dt = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} X(\tau, f) w(\tau - t) e^{j2\pi ft}\, d\tau df. \quad (2)$$

For use in digital signal processing formulas (1) and (2) have to be discretized, using sums instead of integrals and sums of finite length. This is discussed under the term Gabor transformation. The Gabor transformation is defined as follows: From a single prototype or window function $g(t)$, which is localized in time and frequency, a Gabor system $g_{na,mb}(t)$ is derived by time shift and frequency shift as follows (see [19]):

$$g_{na,mb}(t) = e^{2\pi jmbt} g(t - na),\ n, m \in \mathbb{Z},\ a, b \in \mathbb{R}, \quad (3)$$

The time frequency plane is then covered by a lattice of local functions with the lattice constants a time shift and b frequency shift. The Gabor transformation is calculated as sampled STFT of $x(t)$ (see [6]). With the Gabor system $g_{na,mb}(t)$ this can be expressed as follows:

$$c_{nm} = X(na, mb) = \int_{-\infty}^{+\infty} x(t) g^*_{na,mb}(t)\, dt. \quad (4)$$

c_{nm} are called the Gabor coefficients of $x(t)$. The inverse transformation or reconstruction recreates the signal $x(t)$ from its Gabor coefficients. This inverse transformation is called Gabor expansion and is calculated with the window function $\gamma(t)$, which is called dual window of $g(t)$ and which depends on $g(t)$ and on the lattice constants a and b. The Gabor system $\gamma_{na,mb}(t)$ of $\gamma(t)$ is also defined as:

$$\gamma_{na,mb}(t) = e^{2\pi jmbt} \gamma(t - na),\ n, m \in \mathbb{Z},\ a, b \in \mathbb{R}, \quad (5)$$

The inverse transformation is then defined as follows (see [1]):

$$x(t) \sum_{k=-\infty}^{\infty} |\gamma(ka)|^2 = \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} c_{nm} \gamma_{na,mb}(t). \quad (6)$$

The role of analysis and synthesis window is interchangeable.

As already mentioned, the window function $g(t)$ has to be localized in time and frequency. The same is necessary for the dual window $\gamma(t)$. The theory of the Gabor transformation led to the result, that the window functions $g(t)$ and $\gamma(t)$ and the lattice constants a and b must fulfill certain requirements in order to assure the invertibility. This is discussed further in the following two sections.

2.2 Choice of the window function

We start with the choice of the window function. The localization in time and frequency has to be discussed in the context of the uncertainty principle of Heisenberg. It says that the product of the time duration Δt of a window function and its frequency extent Δf has a total lower limit. If Δt and Δf are defined as standard deviation of the window function and its Fourier transformation respectively, this can be expressed with the following inequality (see [18]):

$$\Delta t \Delta f \geq \frac{1}{4\pi}. \quad (7)$$

The "=" is only reached for the Gaussian as window function (see e.g. [20]):

$$g(t) = \frac{1}{\sqrt{2\pi\sigma_t^2}} e^{-\frac{1}{2}\frac{t^2}{\sigma_t^2}}. \quad (8)$$

Its Fourier transformation has the same Gaussian shape as the time function itself:

$$G(f) = \frac{1}{\sqrt{2\pi\sigma_f^2}} e^{-\frac{1}{2}\frac{f^2}{\sigma_f^2}}. \quad (9)$$

With Δt and Δf defined as standard deviation of the time function and its Fourier transformation respectively ($\Delta t = \sigma_t$, $\Delta f = \sigma_f$) and formula (7) for the "=" we get:

$$\sigma_f = \frac{1}{4\pi\sigma_t}. \quad (10)$$

A Gaussian window has the following properties:

- Minimal extent in the time frequency plane according to the Heisenberg uncertainty principle.
- Localized shape, i.e. only one local and global maximum and strict decay in time and frequency direction.

This localization properties are of essential importance for audio processing, in general and audio editing, in specific, because the reference sensor, the human ear itself has a time frequency resolution, which closely reaches the physical limit expressed in the Heisenberg principle (see [2]). It is furthermore capable of adapting its time-frequency resolution in accordance to the Heisenberg principle. The Gaussian as window function leads to Gabor coefficients, which have the best localized influence possible according to the Heisenberg principle. As the Gabor coefficients represent the signal similar or better localized in time and frequency as the human ear, it is possible to avoid perceivable artefacts introduced through editing. The meaning of the Heisenberg principle for Audio Brush is discussed further in chapter 3.

2.3 Choice of dual window function, lattice constants and oversampling factor

With the choice of the window-function, it is still an open issue, which dual window-function $\gamma(t)$ and lattice constants

a, b to choose. As these are interconnected issues, which lead additionally to the question of oversampling, they are discussed together in this section.

The dual window $\gamma(t)$ should also be localized in time and frequency to preserve the local influence of the Gabor coefficients on the result of the inverse transformation. This is of course best achieved by the Gaussian as dual window-function. Gabor in his original paper ([8]) has not discussed the question of the dual window. He suggested to choose $ab = 1$ which is called critically sampled. This choice has implicit influence on the shape of the dual window. In fact with the Balian-Low theorem it can be shown, that in this case, the dual window extends to infinity (see [7]) and is not localized. Due to the interchangeable role of analysis and synthesis window either the analysis or the synthesis is numerically instable. The solution is to choose $ab < 1$, which is referred to as the oversampled case and leading to better localized dual windows and numerically stable analysis and synthesis.

We therefore have to determine an appropriate oversampling factor for $ab < 1$. In the literature normally the cases of rational oversampling ($ab = \frac{p}{q}$, $p, q \in \mathbb{N}$ and $p < q$) and integer oversampling ($ab = \frac{1}{q}$, $q \in \mathbb{N}$) are discussed. Bastiaans [1] proposes to take $ab = \frac{1}{3}$ for which the ideal dual window of the Gaussian is getting very close to a Gaussian window. He mentions that for increasing values of q the resemblance of the Gaussian window and its dual window further increases. By simple empirical hearing tests we found that an oversampling factor of $ab = \frac{1}{5}$ is needed to avoid hearable differences between the original and reconstructed sound in case of using the Gaussian window for analysis and synthesis[2].

In the following if necessary, we name a, b for the critical sampled case a_{crit}, b_{crit} and for the oversampled case a_{over}, b_{over}. The resulting lattice and how it covers the time-frequency plain is illustrated in figure 4. The gray shaded circles indicate the extent of a single Gaussian window in the time-frequency plain expressed in its standard deviations σ_t and σ_f. To ensure the same overlapping of the Gaussians in

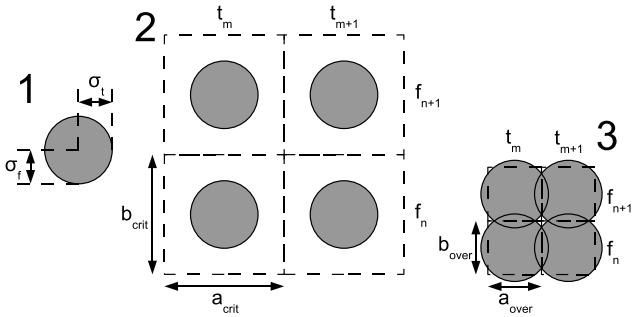


Figure 4: Coverage of the time-frequency plain: The gray shaded circles indicate the extent of a single Gaussian window expressed in its standard deviations σ_t and σ_f (1) for the critical sampled case (2) and the oversampled case (3).

[2]Sound examples showing the influence of the oversampling factor can be found at: http://www.informatik.uni-augsburg.de/~boogaart/..

time and frequency direction, we therefore have to set:

$$\frac{\sigma_t}{\sigma_f} = \frac{a_{crit}}{b_{crit}} = \frac{a_{over}}{b_{over}}. \tag{11}$$

With $a_{over} b_{over} = \frac{1}{q}$ this holds for:

$$a_{over} = \frac{a_{crit}}{\sqrt{q}}, b_{over} = \frac{b_{crit}}{\sqrt{q}}. \tag{12}$$

With formula (10) and formula (11) we can solve:

$$\sigma_t = \frac{1}{\sqrt{4\pi}}\sqrt{\frac{a}{b}}, \sigma_f = \frac{1}{\sqrt{4\pi}}\sqrt{\frac{b}{a}}. \tag{13}$$

With these formulas and $q = 5$, we still have the freedom to choose either a_{crit} or b_{crit} and can calculate all other necessary values. We can than use the Gaussian window $g(t)$ as its own dual window $\gamma(t) = g(t)$.

2.4 Implementation issues

2.4.1 Discretization

The formulas (4) and (6) are for the continuous case of $x(t)$. In the time discrete case $x(t)$ is sampled with the sampling frequency f_s. With $T = 1/f_s$ and $k \in \mathbb{Z}$ $x(t)$ becomes $x(kT)$. To fulfill the sampling theorem ([14]), the bandwidth f_B of $x(kT)$ has fo fulfill $f_B \leq f_s/2$. For HiFi audio typical values are $f_s = 44.1kHz$ (CD) or higher and $f_B = 20kHz$.

For discretized signals the Gabor transformation (4) gets the form:

$$c_{nm} = \frac{1}{L}\sum_{k=-\infty}^{\infty} x(kT) g^*_{na,mb}(kT) \tag{14}$$

and the inverse Gabor transformation (6) gets the form:

$$x(kT) = \frac{1}{L}\sum_{n=-\infty}^{\infty}\sum_{m=0}^{M} c_{nm} g_{na,mb}(kT) \tag{15}$$

with

$$L = \sqrt{q\sum_{k=-\infty}^{\infty}|g(ka)|^2} \tag{16}$$

and

$$m \in [0, M] \text{ and } M = \lceil f_B/b \rceil. \tag{17}$$

Remark: Storing the Gabor coefficients is very effective. A discretized time signal of length $t_{duration}$ needs for storage

$$N^{\mathbb{R}}_{samples} = t_{duration} f_s \tag{18}$$

real values. The Gabor coefficients need

$$N^{\mathbb{C}}_{GaborCoeff} = \frac{t_{duration}}{a}\frac{f_B}{b} = t_{duration} f_B q \tag{19}$$

complex or

$$N^{\mathbb{R}}_{GaborCoeff} = 2 t_{duration} f_B q \tag{20}$$

real and imaginary values combined for storage, i.e.

$$N^{\mathbb{R}}_{GaborCoeff} \leq q N^{\mathbb{R}}_{samples} \tag{21}$$

independent of the time-frequency resolution.

2.4.2 Truncation

The formulas (14) and (15) still use a sum over infinite time. To implement them a truncation of the sum is necessary, which corresponds to a truncation of the windows. To find an appropriate truncation we have chosen the way of simple empirical listening tests again, with the goal to get a reproduced sound with no hearable difference from the original sound. The decline D of the Gaussian window from the maximum to the cut is expressed in dB:

$$D = 20log\frac{g(0)}{g(t_{cut})}dB. \tag{22}$$

For a given D we get with (8):

$$t_{cut} = \sqrt{2ln\left(10^{\frac{D}{20}}\right)}\sigma_t. \tag{23}$$

The digital implemental form of the Gabor transformation is then:

$$c_{nm} = \frac{1}{L}\sum_{k=-N}^{N} x(kT)g^*_{na,mb}(kT) \tag{24}$$

and the inverse:

$$x(kT) = \frac{1}{L}\sum_{n=-N}^{N}\sum_{m=0}^{M} c_{nm}g_{na,mb}(kT) \tag{25}$$

with:

$$L = \sqrt{q\sum_{k=-N}^{N}|g(ka)|^2} \tag{26}$$

and

$$N = t_{cut}f_s = \frac{t_{cut}}{T}, m \in [0, M]\text{and}M = \lceil f_B/b_{over}\rceil. \tag{27}$$

In our implementation high values for D (up to $200dB$) have been tested in $10dB$ increments, but values of $D \geq 30dB$ have shown to be completely sufficient for high quality audio[3].

3. TIME-FREQUENCY ZOOMING

3.1 Time-frequency resolution

As mentioned in section 2.2, the human ear usually adapts its current time-frequency resolution to the current content of the signal according to the Heisenberg uncertainty principle. It is therefore advantageous also to adapt the resolution of our transformation to the current editing task.

3.1.1 Heisenberg Uncertainty Principle

We already applied the Heisenberg uncertainty principle to the functions of the window (see section 2.2) and the dual window (see section 2.3). It also holds for the function of the signal, which of course in general has a resolution worse than the theoretical limit.

As the Gabor transformation is a discretized version of the STFT, we can discuss the continuous case of the STFT.

[3]Sound examples showing the influence of the truncation of the Gaussian window can be found at: http://www.informatik.uni-augsburg.de/~boogaart/.

The STFT (see eq. (1)) can be expressed as a convolution of signal $x(t)$ with $h^*(-t, f) = w^*(-t)e^{-j2\pi ft}$:

$$X(t, f) = x(t) * h^*(-t, f) = \int_{-\infty}^{+\infty} x(\tau)h^*(-(t-\tau), f)\, d\tau. \tag{28}$$

This is similar to adding the variances of two statistically independent random variables X and Y, which form a new random variable $Z = X + Y$. Their probability density functions are also convolved and the variance of Z is then given by (see [10]):

$$\sigma_Z^2 = \sigma_X^2 + \sigma_Y^2. \tag{29}$$

Therefore the time and frequency variances of a signal and a window are added in the form:

$$\sigma^2_{t_{transformation}} = \sigma^2_{t_{signal}} + \sigma^2_{t_{window}}, \tag{30}$$

$$\sigma^2_{f_{transformation}} = \sigma^2_{f_{signal}} + \sigma^2_{f_{window}}. \tag{31}$$

Consequently the achievable accuracy of editing a given signal is given by the superposition of the signal's and window's uncertainty. As time and frequency resolution are interconnected, one has to give up time resolution in order to improve the frequency resolution and vice versa. Choosing an adapted window length allows locally to minimize the influence of the window on the representation of a signal in the spectrogram.

3.1.2 Multiwindow Analysis

With the choices of chapter 2 ($g(t) = \gamma(t)$ gaussian, $p = 1$, $q = 5$, $D = 30dB$), we can still choose the frequency shift b_{crit} resulting in a time shift $a_{crit} = \frac{1}{b_{crit}}$ or vice versa. This is equivalent to choosing an adapted window length. Different choices lead to different time frequency representations of the same signal in the 3D-space with the axes t, f and e.g. b_{crit}. However, the same signal is represented and thus this space is overcomplete: different characteristics of the signal are revealed in different layers with constant b_{crit}. The properties of this space can be clarified by the extremes of b_{crit}:

For $b_{crit} \to \infty$: The window $g(t)$ becomes the Dirac impulse and the Gabor transformation becomes the time signal itself.

For $b_{crit} = 0$: The window becomes $g(t) = const.$ losing its windowing properties and the Gabor transformation becomes the Fourier transformation.

It is therefore proposed to calculate the Gabor transformation of an audio signal with different choices of b_{crit} and to perform the respective editing task in the layer $b_{crit} = const.$ which allows the best accuracy for the current task. This can be understood as zooming, which allows to increase the resolution of the representation of a signal either in time or in frequency. As result of the Heisenberg uncertainty principle, the resolution of the other domain always decreases.

3.2 Example

Figure 5 shows a typical spectrogram of a speech signal. The spectrogram is calculated with $b_{crit} = 64Hz$, which results with $b_{over} = 28.6Hz$ in roughly 699 frequency bands in the range from 0 Hz to 20 kHz. Figure 6 for comparison

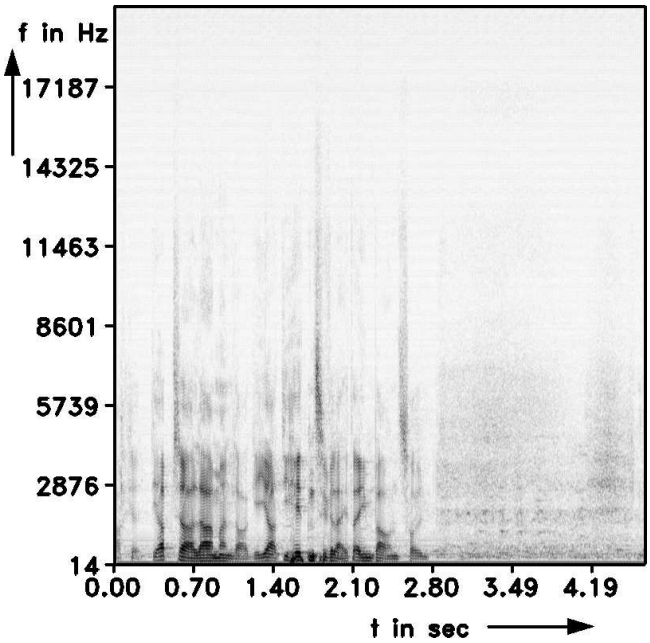


Figure 5: Typical spectrogram for speech calculated with $b_{crit} = 64Hz$, i.e. $b_{over} = 28.6Hz$.

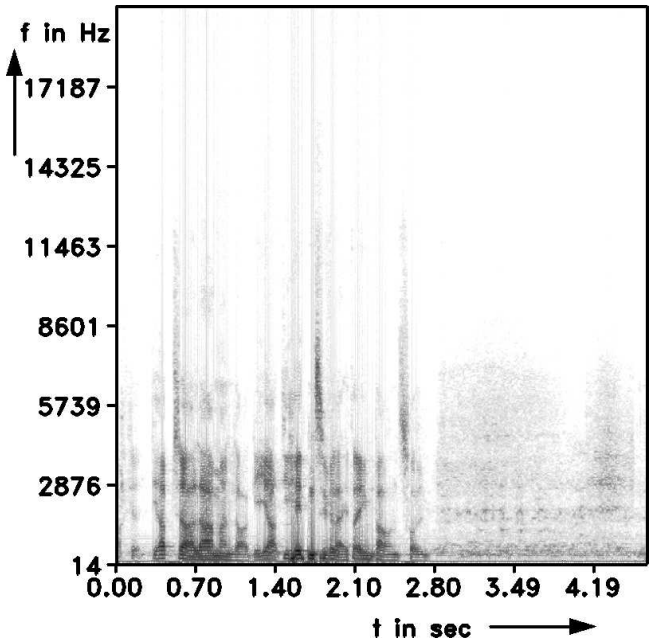


Figure 6: Spectrogram for speech calculated with FFT of blocklength 1024 samples.

shows the same signal calculated with a FFT of 1024 samples blocklength which at the sampling frequency of 48 kHz results in approximately 427 frequency-bands in the range from 0 Hz to 20 kHz. A rectangular window is applied, to preserve the invertibility. The Gabor spectrogram contains no noise effects from windowing, window truncation or similar. In contrast the FFT spectrogram contains artefacts, because of the rectangular window, which preserves the invertibility. Especially at high frequencies, the spectrogram contains vertical lines, which have no justification from the signal. They are introduced by the FFT. Also at lower frequencies, which contain the voiced parts of the speech signal, from 100 Hz to 300 Hz, the Gabor spectrogram is much smoother then the FFT spectrogram.

One property of the signal is hidden in both spectrograms: The recording was accidently interfered by power line hum at 50 Hz (common in Europe). The hum can be heard, if the sound is played at higher volume levels, but it cannot be seen in the spectrograms. Figure 7 a) shows a spectrogram of the same recording with a much higher frequency resolution of $b_{crit} = 5Hz$, i.e. $b_{over} = 2.24Hz$. In this spectrogram it is easy to distinguish the hum at 50 Hz and his higher harmonics from the rest of the signal.

3.3 Comparison of multiwindows and wavelets

The idea of using different window lengths for the same signal is similar to the idea which led to wavelets. High frequencies normally, not only in the field of audio, contain more transient parts of the signal, which can be better revealed with shorter windows. Therefore for high frequencies wavelets use shorter windows, halving the window length for doubled mid-frequencies, i.e. every octave (see [18]). In the following we will discuss why the Gabor transformation combined with the idea of multiwindows is superior to wavelets for the application to high quality audio.

Wavelets have been successfully applied e.g. in speech processing ([12]), which normally is done on a bandwidth from 300 Hz to 3400 Hz. As to be seen in figure 8: The solid black line indicates the critical bandwidth ([22]), which is a measure e.g. for the frequency resolution of the human ear (100 different pitches can be distinguished per bandwidth of one critical band). The dashed black line indicates a common rule of thumb for the critical bandwidth: $f_{cb} = 100Hz$ from $0Hz$ to $500Hz$, independent of the mid-frequency and $f_{cb} = 0.2f$ over $500Hz$. The vertical dashed blue lines indicate the typical bandwidth for speech transmission over a telephone line, which is from $300Hz$ to $3400Hz$. The red line shows bandwidth against mid-frequency for a typical wavelet setting with $bandwidth \propto f$. The solid blue line shows bandwidth against mid-frequency for one window length for the Gabor transformation with $bandwidth = const.$, independent of f. It can clearly be seen that the time-frequency resolution of wavelets is able to emulate that of the human ear in the range from $300Hz$ to $3400Hz$ very closely, which made them very attractive for speech processing. However the opposite is true for the bandwidth of music from $0Hz$ to $20kHz$. If a wavelet transformation is chosen, which is appropriate for high frequencies, the frequency resolution for low frequencies gets too fine, the time resolution gets too bad, if one is chosen, which fits for one low frequency, it is inappropriate for every other frequency and the frequency resolution for high frequencies gets too rough and the time resolution gets too fine. Therefore a multiwindow approach allows possibilities, which neither are possible for wavelets nor can be made possible by a multiwindow-wavelet approach for the high bandwidth needed for high quality audio.

4. AUDITORY BRUSHING

In this chapter we will discuss and give examples for the two most important steps in visual audio editing: (a) de-

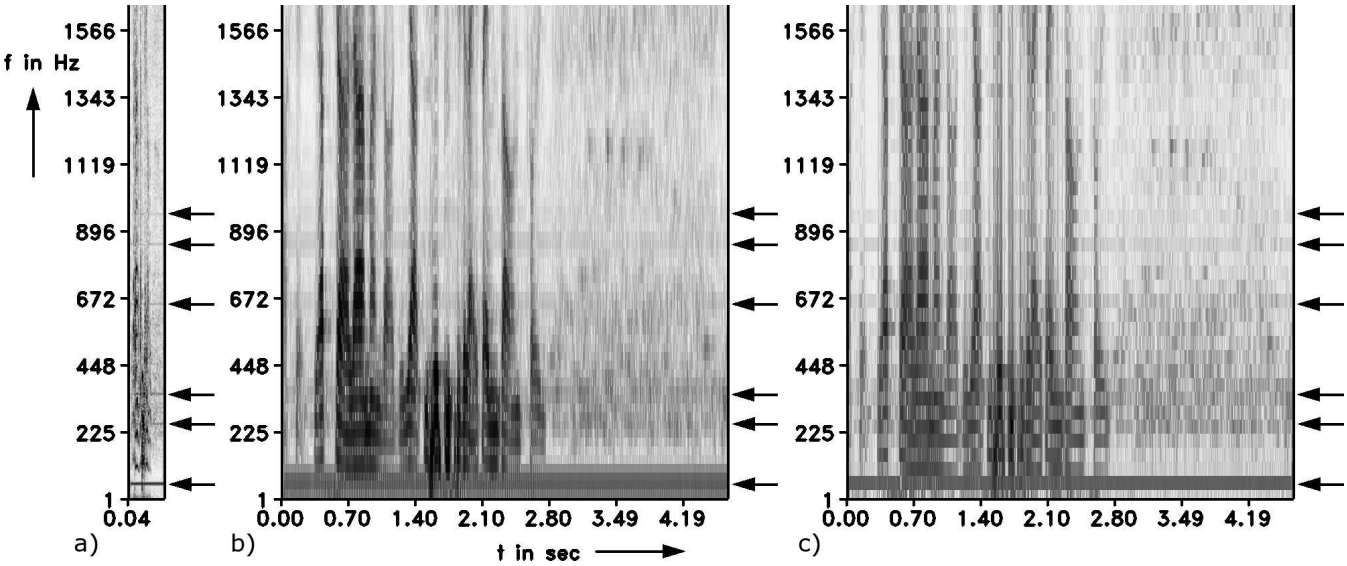


Figure 7: Spectrograms for speech signal. For convenience they are cut above 1600 Hz. a): $b_{crit} = 5Hz$, b): $b_{crit} = 64Hz$, c): FFT with 1024 samples blocklength. The interfering power line hum and higher harmonics can be recognized easily in the left spectrogram. They are indicated by arrows on the right at 50 Hz, 250 Hz, 350 Hz, 650 Hz, 850 Hz and 950 Hz. The energy of the hum is distributed widely in the middle and right spectrogram and the higher harmonics are totally blurred.

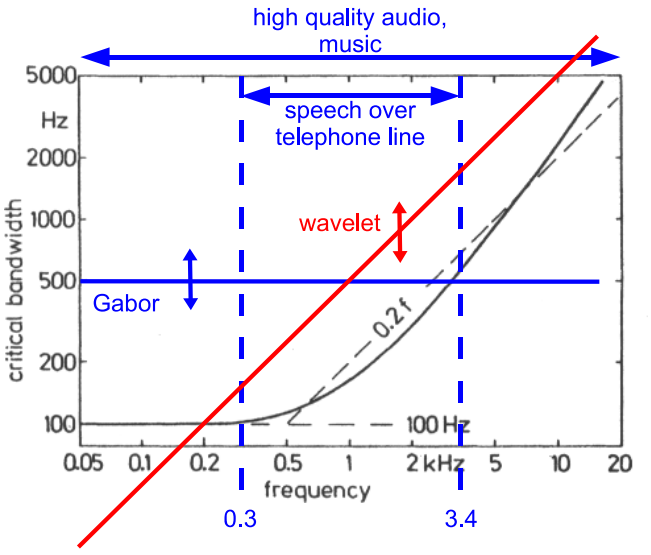


Figure 8: Solid black line: critical bandwidth of human ear against mid-frequency. Dashed black line: common rule of thumb for the critical bandwidth. Vertical dashed blue lines: typical bandwidth for speech transmission over telephone line. Red line: typical wavelet setting, $bandwidth \propto f$. Solid blue line: Gabor transformation, $bandwidth = const.$. The black parts of the image are taken from [22].

tection of known audio objects and (b) removal of detected audio objects from the original audio track.

Detections are described by the locations (i.e. positions and shapes) of the audio objects in the spectrograms. Detection will have to be performed resilient to typical variations in which audio objects of interest can be experienced, and this will be discussed in detail in section 4.1.

Audio object removal is a special form of modifying audio objects. An example for a more general modification is the application of a sound equalizer to only the detected audio objects, while at the same time leaving the remaining signal content unchanged. Such kind of modifications will be discussed in section 4.2. Figure 9 summarizes these two possible two-step editing processes.

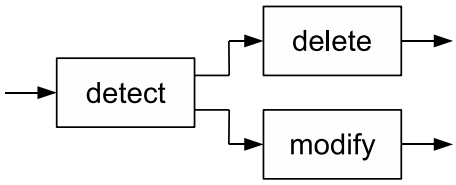


Figure 9: Two-step editing process for audio objects.

Remark: The data in the time-frequency plane is represented by complex values as real and imaginary 32 bit float values. The processing tasks are performed on the magnitude values, while the phases are stored for the inverse transformation. The visualization uses the magnitude values, which are compressed by calculating the square root and quantizing to 8 bit unsigned integer values.

4.1 Detecting audio objects

Certain reproducible sounds such as individual notes played by a piano or any other instrument, whistles, and other

well-structured sounds can be treated as visual objects in the spectrogram. In there, they are characterized by a distinctive visual pattern - a pattern which looks similar even under typical variations such as frequency shifts, different play rates, and recordings from different microphones, different rooms, and playback devices.

Detection starts with a template of an audio object. Figure 10 explains the procedure. The audio object template

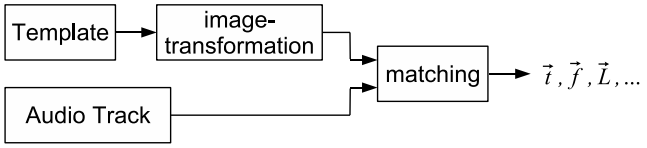


Figure 10: Detecting audio objects.

undergoes a set of predefined parameterized image-transformations[4] before a visual matching algorithm is used to possibly detect the modified template in the audio track.

4.1.1 Sound variations, image-transformations

As mentioned before, each audio template is transformed before matching, to compensate for the different setting used for recording the template and the audio track (different microphones, different rooms, different instrument of the same kind, etc.). This results in slightly different spectrograms, different levels and sampling rate differences. While for the first and the second, the detection relies on a robust matching algorithm, the third can be avoided by the preprocessing step called image-transformation, see Figure 10.

The sampling rate differences have the following impact on the template or the audio track:

- A higher sampling rate results in more samples in a given time. Compared to the correct sampling rate this results in a longer image. The template has to be compressed in time direction to compensate for this.
- A higher sampling rate recording played at the correct sampling rate results in a lower sound. The template has to be stretched in frequency direction, to compensate for this.
- Regardless the sampling rate difference, which is unknown, a compression in one direction is connected to stretching in the other. The template therefore undergoes a combined stretching and compression.

4.1.2 Matching

The visual matching algorithm is used with each possible set of allowed image-transformation parameters in order to find the best matches in the spectrograms under the allowed image-transformation space. As a result, a vector of locations (time, frequency, and shape) and perhaps other parameters such as volume level and alike are given wherever the template has been detected in the audio track.

In our system we use the normalized cross-correlation between the modified audio template image and all possible locations in the spectrogram. As a result, audio object are usually located with pixel accuracy. Since audio editing is very sensitive to inaccuracies, the estimated locations and the associated transformation parameters of the reference (i.e. template) audio object could be further refined by the Lucas-Kanade feature tracker ([13]) – an optical flow method – leading to subpixel accuracy. This technique for example is used in conjunction with sub-pixel accurate feature point localization in camera calibration (see [11]).

[4]These transformations, discussed in the image domain, are named image-transformation, to distinguish them from the time-frequency transformations in this paper.

4.2 Deleting audio objects

An audio object, which has been detected in an audio track, can now be edited by the use of the template which matches best. Possible editing tasks are: correcting the level, applying a kind of equalization oder completely deleting. In this section, two different methods for deleting audio objects are mentioned.

4.2.1 Stamping

The first approach simply "stamps" the template out, i.e. the magnitude values are set to zero. Either a user decides, by inspecting the spectrograms visually and the result aurally, or the template is applied automatically. All magnitude values, which in the template spectrogram are larger than a certain frequency dependent threshold value are stamped in the audio track (see figure 11).

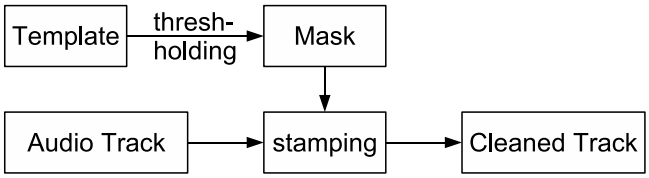


Figure 11: Scheme for stamping a detected audio object with a template spectrogram.

We apply this approach to a mixed music and whistle signal. Figure 12 shows the relevant signals and spectrograms: a) music signal, b) whistling signal, c) mixed signals. Both signals can be recognized in the spectrogram and the time delay of the whistling of 4 sec can be determined easily. Figure 12, d): "cleaned" signal: the whistling is stamped out with the template spectrogram, which was created from a different recording of the same whistle signal. e) overcompensated signal parts and f) under-compensated signal parts. In the cleaned signal, the whistling is hardly perceivable and the speech signal has no perceivable difference to the uninterfered original [5]. The results could perhaps be improved further by the following two methods: First: Convolving the stamping mask with a Gauss-edged smoothing kernel, with the standard deviation of the window $g(t)$ of the current layer (see section 3.1.1). Second: Filling the wholes in the spectrogram with appropriate image inpainting algorithms (see [3]).

4.2.2 Energy based erasing

In contrast to visual objects, which are often intransparent, audio objects always shine through the energy of an other audio object. The stamping approach, although attractive because of its simplicity and analogy to the visual domain, creates poorer results for increased overlapping of objects in the time frequency domain.

[5]See http://www.informatik.uni-augsburg.de/~boogaart/ for the sound examples in wav-format.

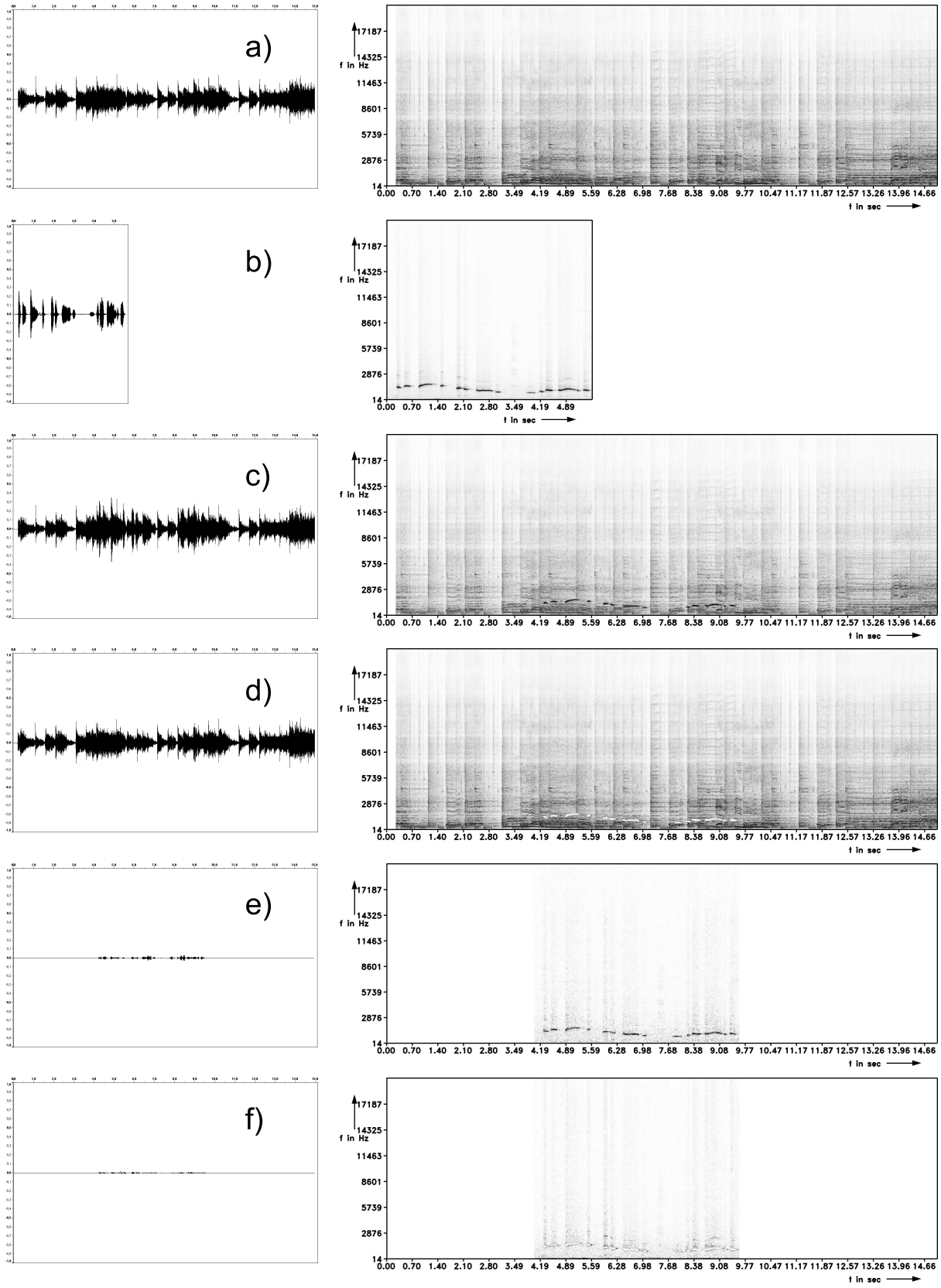


Figure 12: From top to bottom a) music signal, b) whistling signal, c) mixed signals, whistling 4 sec delayed, d) stamped signal, i.e. cleaned signal, e) over-compensated signal parts and f) under-compensated signal parts.

Another method is to subtract the magnitude values of the templates spectrogram from the magnitude values of the mixed signals spectrogram. As the template was recorded with a different microphone and perhaps has a different level, it is first adapted in order to match the mixed signals spectrogram absolutely and per frequency band as well as possible, before applying the difference. Figure 13 shows a scheme for erasing an audio object by subtracting the energy of an adapted template spectrogram. The success of this method depends on the similarity of template and original signal and the adaption of the template to the original signal respectively[6].

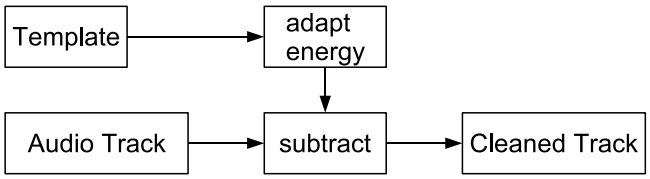


Figure 13: Scheme for erasing a detected audio object with a template spectrogram by subtracting the magnitude values.

5. CONCLUSION

A new tool for audio editing has been presented: the Audio Brush. It allows editing audio in an intuitive and direct way. The freedom to choose a window length for forward and inverse transformation is discussed, which opens the possibility, of using an optimal time-frequency resolution in order to reveal certain properties of a signal.

Some methods were presented for detecting and deleting audio objects. They show the benefit of the new approach. As the results are not perfect, this is not because of the forward and inverse transformation proposed, but because of the imperfect brushes applied. Therefore further research has to enhance to auditory image processing methods needed, in order to edit the magnitude values as accurate in magnitude as they can be perceived by the human ear.

Possible further applications of the Audio Brush are: Improving the live recording of an instrument: All notes are deleted and replaced by a studio recorded version, e.g. to correct the tune of the instrument. An other application might be to enhance old video material, containing e.g. a speech. Given the speech is interfered by some music played in the background, the music could be deleted in the spectrogram of the audio track.

6. REFERENCES

[1] M. J. Bastiaans. Optimum sampling distances in the gabor scheme. *Proc. CSSP-97, Mierlo, Netherlands*, pages 35–42, 1997.

[2] C. G. v. d. Boogaart. *Master thesis: Eine signal-adaptive Spektral-Transformation für die Zeit-Frequenz-Analyse von Audiosignalen*. Technische Universität München, 2003.

[3] A. Criminisi, P. Perez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on Image Processing*, 13:1200 – 1212, 2004.

[4] I. Daubechies. *Ten lectures on wavelets*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1992.

[5] D. Donoho. Nonlinear wavelet methods for recovery of signals, densities, and spectra from indirect and noisy data. In *Different Perspectives on Wavelets*, volume 47 of *Proceeding of Symposia in Applied Mathematics*, pages 173–205, 1993.

[6] H. G. Feichtinger and T. Strohmer. *Gabor Analysis and Algorithms: Theory and Applications*. Birkhäuser, Boston, 1998.

[7] H. G. Feichtinger and T. Strohmer. *Advances in Gabor Analysis*. Birkhäuser, Boston, 2003.

[8] D. Gabor. Theory of communication. *Journal of the Institution of Electrical Engineers*, pages 429–457, November 1946.

[9] J. Haitsma and T. Kalker. A highly robust audio fingerprinting system. *Third International Conference on Music Information Retrieval*, pages 107–115, 2002.

[10] J. Hartung, B. Elpelt, and K. Klösener. *Statistik. Lehr- und Handbuch der angewandten Statistik*. Oldenbourg, München, 1995.

[11] E. Hörster, R. Lienhart, W. Kellerman, and J.-Y. Bouguet. Calibration of visual sensors and actuators in distributed computing platforms. *3rd ACM International Workshop on Video Surveillance & Sensor Networks*, November 2005.

[12] B. B. Hubbard. *The World According to Wavelets: The Story of a Mathematical Technique in the Making*. A K Peters, Wellesley, Massachusetts, 1996.

[13] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, pages 674–679, April 1981.

[14] H. Marko. *Systemtheorie*. Springer-Verlag, Berlin, 1995.

[15] A. V. Oppenheim and R. W. Schafer. *Discrete-time signal processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989.

[16] S. Qian and D. Chen. *Joint time-frequency analysis: methods and applications*. Prentice Hall, New Jersey, 1996.

[17] D. A. Reynolds. An overview of automatic speaker recognition technology. In *Proc. ICASSP*, volume 4, pages 4072–4075, 2002.

[18] O. Rioul and M. Vetterli. Wavelets and signal processing. *IEEE Signal Processing Magazine*, 8(4):14–38, 1991.

[19] T. Strohmer. Approximation of dual gabor frames, window decay, and wireless communications. *Appl.Comp.Harm.Anal.*, 11(2):243–262, 2001.

[20] P. A. Tipler. *Physik*. Spektrum Akademischer Verlag, Heiderlberg, Berlin, 1994.

[21] P. J. Wolfe, S. J. Godsill, and M. Dörfler. Multi-gabor dictionaries for audio time-frequency analysis. *Proceedings of WASPAA 2001*, 2001.

[22] E. Zwicker and H. Fastl. *Psychoacoustics. Facts and Models*. Springer Verlag, second updated edition, 1999.

[6]See http://www.informatik.uni-augsburg.de/~boogaart/ for an audio example demonstrating this method.