Saarland University
Faculty of Natural Sciences and Technology I
Department of Computer Science
Master's Program in Visual Computing

Master's Thesis

Can Image Processing Help Music Information Retrieval?

Enhancing Spectrograms for Feature Extraction

Jan Hendrik Dithmar

2010-7-4

Supervisor PD Dr. Martin Welk
Advisor Peter Grosche, M.Sc.
Reviewers PD Dr. Martin Welk
PD Dr. Meinard Müller

Affidavit

I hereby declare that this master's thesis has been written only by the undersigned and without any assistance from third parties.
Furthermore I confirm that no sources have been used in the preparation of this thesis other than those indicated in the thesis itself.

Saarbrücken, 2010-7-4

Declaration of Consent

Herewith I agree that this thesis will be made available through the library of the Computer Science Department. This consent explicitly includes both the printed, as well as the electronic form attached to this thesis.
I confirm that the electronic and the printed version are of identical content.

Saarbrücken, 2010-7-4

To Nellie and Noah

Abstract

In today's world, a lot of information is accessible in a digital form. This is especially true for images or music. To make use of this variety of data, it has to be processed meaningfully which leads to the task of information retrieval. The field of music information retrieval (MIR) allows the user to access and explore music. For example, the user could get additional information about melodic and harmonic structures or rhythm and tempo. This is fruitful information for music analysis. Another application is to find a convenient way to navigate inside a piece of music. This could be realized by searching for structures like similar rhythmic patterns or similar chord progressions and giving the user the opportunity to jump directly to certain similar parts. However, such purely content-based tasks are very tough.

For the context of this thesis, we focus on chord recognition and onset detection and try to enhance spectrograms that are computed from the music data. For chord recognition, we have to enhance frequency contributions that belong to the notes in a chord at a certain time. In contrast, in the context of onset detection it is important when an onset occurs, not which frequencies contribute to it. In other words, vertical structures have to be enhanced for onset detection.

We are looking at the different problems mainly from an image processing viewpoint. We apply several image processing methods to spectrograms where we want to investigate whether we can improve the quality of the spectrogram for further processing by feature extraction algorithms.

Acknowledgements

I would like to express my sincere thanks to PD Dr. Martin Welk for the excellent mentoring and support during the development of this thesis. His encouragement and our vivid discussions as well as the friendly atmosphere in the Mathematical Image Analysis Group greatly assisted me in my work. Furthermore, I gratefully acknowledge that Prof. Dr. Joachim Weickert provided me with a topic that fits both my interests in image processing and in music. I also like to thank PD Dr. Meinard Müller for the cooperation with the Mathematical Image Analysis Group (which makes it possible for me to work on this topic) and for consenting to review my thesis. Many thanks are also devoted to Peter Grosche for giving me very good insights into music processing and for the great support concerning MATLAB.

Special thanks go to my family who enabled me to perform my studies and to pursue my musical career as I did and who therefore also significantly contributed to the success I have enjoyed so far.

Contents

Chapter 1

Introduction

A journey of a thousand miles begins with a single step.
— Lao-tzu

1.1 Motivation

In today's world, a lot of information is accessible in a digital form. This is especially true for images or music. Digital music is present in our everyday life and the possibilities to handle digital music enhanced dramatically. The field of *music information retrieval* (MIR) allows the user to access and explore music. But why are we interested in music information retrieval? For example, the user could get additional information about melodic and harmonic structures or rhythm and tempo. This is fruitful information for music analysis. Another application is to find a convenient way to navigate inside a piece of music. This could be realized by searching for structures like similar rhythmic patterns or similar chord progressions and giving the user the opportunity to jump directly to certain similar parts. However, such purely content-based tasks are very hard. This is due to the complexity of music data.

Among the many challenges that arise with information retrieval in a piece of music, we will focus on *chord recognition* and *onset detection* that are further discussed in Chapter 2.

In image processing, there exists a wide variety of challenging tasks, such as face recognition, image denoising or problems in optic flow, to name just a few. Image denoising is concerned with enhancing the visual quality of an image as much as possible. Practical tasks in image denoising are e.g. in medical imaging such as MRT or ultrasound.

An interesting question is what do both worlds have in common? Image processing and music processing use the Fourier transform [6] as an important tool. Furthermore, the concept of noise is also present in both settings whether the noise is global like Gaussian noise or locally like a black spot in a color image or a cough in a piece of music. Looking deeper in both approaches we can even find types of "noise" that are present (e.g. due to the acquisition technique) but are not necessarily perceived as such. This could be in a digital image due to the sensor of the camera or in music due to the sound produced by the mechanics of an instrument. The noise produced by a camera sensor could lead to a deterioration of colors in

an image while the sound of the instrument mechanics contributes to every frequency in the spectrogram making applications in music processing probably much harder. Moreover, the spectrogram representation (compare top of Figure 2.1) gives us – from an image processing perspective – a convenient and familiar way to access the audio data. Thus, we have strong evidence that image processing methods can also be applied to music data.

In this thesis, we are looking at the problems mainly from an image processing viewpoint. We apply several image processing methods to spectrograms computed from the music data where we want to investigate whether we can improve the quality of the spectrogram for further processing by feature extraction methods.

Since we are investigating the topic from an image processing viewpoint, we also use the terminology common in this field where terms to describe certain concepts – such as "noise" – might probably differ. An in-depth introduction to music data processing is given by Meinard Müller in [25].

1.2 Goals

In this thesis, we want to evaluate how image processing methods can contribute to the field of music processing and analysis. We concentrate on enhancing spectrograms computed from the music data such that the feature extraction process necessary for every MIR task will be more convenient. Since the spectrograms are a graphical representation of the music data, algorithms that are well understood in the image processing context, such as diffusion processes [32] or non-local means [8] can be applied to the spectrograms.

To judge whether image processing algorithms help in this context, we mostly use them in their basic form, meaning that no improvements or simplifications based on the knowledge of music processing are involved. As a consequence, runtime is not the main concern. Nevertheless, from a practical viewpoint, short response times (ideally, close to interactive behavior) are desirable. Therefore, efficiency aspects will not be completely outside consideration, and we will favor more efficient solutions wherever this is possible without serious compromises in quality. In particular, it is often helpful to adapt the algorithms in a way that built-in fast matrix operations of MATLAB can be exploited.

We will investigate real-world example such as songs by The Beatles or classical music like Beethoven's Fifth.

1.3 Contents

The structure of this thesis is as follows. Chapter 2 gives a short introduction to chord recognition and onset detection as well as a common quality measure used in this context. Chapters 3, 4 and 5 introduce the methods that are used in the experiments, i.e. diffusion processes, morphological operations and NL-means, respectively, where Chapter 6 gives insights in how these algorithms can be modified to adapt to the setting of chord recognition or onset detection. Chapters 7 and 8 give a detailed overview of the experiments where different parameter

settings are compared as well as how good the methods perform on the problem set. Chapter 9 concludes the thesis and gives ideas for future work. An overview of the testfiles used in this thesis is given in Appendix A.

Chapter 2

Tasks in Music Information Retrieval

If a composer could say what he had to say in words
he would not bother trying to say it in music.
— Gustav Mahler

Among others, chord recognition [20, 29, 10] and onset detection [21, 13, 12, 14, 17] constitute topics of active research. They are important tasks in music processing and music information retrieval (MIR) and are very helpful for the analysis of music data.

The first step in every music processing task – and thus in music information retrieval – is to extract suitable features to obtain key aspects while ignoring irrelevant ones. We shortly touch some key aspects of the feature extraction in the corresponding paragraph (2.1 and 2.2).

There exist different visual representations of the audio data, such as a *chroma-time representation*. We do not introduce these here in detail. Instead, the interested reader is referred to the book by Meinard Müller [25]. Nevertheless, the basic intuition of these representations is made clear.

Throughout the thesis, we focus on the *spectrogram representation*.

2.1 Chord Recognition

A musical chord is a set of simultaneous tones. Looking at the chords over time, we can analyze the harmonic structure in a piece of music. Once the harmonic structure is known, a sequence of chords can give insights on higher-level structures such as phrases. The harmonic structure is also strongly related to perceived emotion. Thus, similar chord sequences can be observed in songs that are close in genre or emotion. Automatic chord labelings are very useful for people who want to do harmonic analysis of music and in applications like music search.

The basic idea behind chord recognition is to inspect what frequencies are present at a certain time and match these to music notes. To achieve this, the audio signal is decomposed into spectral bands where each band corresponds to a pitch of the equal-tempered scale. Figure 2.1 shows how such representations could look like. The first image gives a *frequency-time representation*. Each row in a frequency-time representation stands for a frequency in Hz. From

this representation we can go to a *pitch-time representation*. In contrast to the frequency-time representation, the pitch-time representation shows the different MIDI pitches corresponding to the frequencies on the y-axis, instead of the frequencies themselves. A chroma-based representation can be easily obtained from a pitch representation by adding up the subbands that correspond to the same pitch class, meaning that the chroma-time representation accumulates all occurrences of C, C$^\sharp$, D, D$^\sharp$ and so on in one row per pitch class. Please note that we are assuming the equal-tempered scale as used in Western music, meaning that e.g. the notes D$^\sharp$ and E$^\flat$ are considered the same. In other words, by use of the frequency-time representation we are able to extract the pitches that are present at a certain time in a piece of music. Thus, it is crucial that we can clearly distinguish between different frequencies. To achieve this, we have to emphasize horizontal structures. Note that emphasizing vertical structures would result in a blurring across frequency bounds.

Chord recognition comes with some challenges. The biggest challenge is rhythmic accompaniment in every form, be it drums or other percussion, because the sounds produced by drums or percussion contribute to every single frequency. Let us assume that the third of a chord is somehow noisy. This would prevent us from deciding between Major and Minor or – even worse – would lead to a wrong decision.

A description of template-based chord recognition can be found in Konz et al. [19].

2.2 Onset Detection

Also an interesting task in music processing is onset detection. At first glance, this seems to be a well-defined task: the aim is to find the starting time of each musical note. In contrast to chord recognition, onset detection relies on vertical structures in the spectrogram. In other words, we are interested in a good isolation of events for each point in time (i.e. *when* a contribution happens) where it is not important to know *which* frequencies make a contribution. As a consequence, the goal is to enhance vertical structures.

In polyphonic music, where nominally simultaneous notes or chords might spread over a few tens of milliseconds, we see that there is some blurring involved. Another difficult scenario is whenever we have fluent note transitions which is often the case for classical music dominated by string instruments. These problems are depicted in Figure 2.2. For instruments with long attack times it is difficult to define an unambiguous and precise onset time for the notes they produce.

The presence of vertical structures can be seen very well in the top spectrogram of Figure 2.2, especially in the time frame from the beginning until approximately 3 seconds. But what about the example on the bottom of Figure 2.2? There we don't visually recognize any sharp boundaries. And in fact, there are no sharp boundaries. Instead, listening to this excerpt reveals that there is a melody which flows "above" the rhythmical accompaniment that acts like some sort of blur. Due to this fact, it is even difficult at some point to acoustically judge where an onset lies. The *novelty curves* of these two examples are given in Figure 2.3. A novelty curve is a representation that depicts the changes in energy and is computed e.g. by analyzing the signal's spectral content. As a consequence, it is a suitable indicator for note

onsets. A more detailed introduction to novelty curves can be found in [16]. In Figure 2.3 we clearly see that in the second example (on the bottom) the novelty curve changes a lot. If we look at e.g. what happens around the time 5 seconds and in the time frame from about 8 seconds to 10 seconds and compare this with the manually annotated ground truth (red lines), we notice a lot of energy fluctuations – in other words onsets – where actually no onsets should be.

This is a perfect example that the onset detection problem is by far not easy and not well-defined. Further challenges are provided in [22].

Onset detection is an essential step towards advanced tasks like tempo analysis, beat tracking, music synchronization and automatic transcription as well as non-linear time-scaling and pitch-shifting. Due to the fact that music is event-based, the segmentation into individual note events greatly helps in editing and analyzing audio.

2.3 Quality Measure

In the case that there is a ground truth available, we can measure the quality of a processing step by investigating the quantities *precision*, *recall* and *F-measure*, see also [3]. We introduce the precision P as

$$P = \frac{|\{\text{relevant}\} \cap \{\text{retrieved}\}|}{|\{\text{retrieved}\}|}, \tag{2.1}$$

and the recall R as

$$R = \frac{|\{\text{relevant}\} \cap \{\text{retrieved}\}|}{|\{\text{relevant}\}|}. \tag{2.2}$$

Note that precision measures how specific the detection is, while recall measures its exhaustiveness. None of these aspects can be neglected in the quality measurement. To give nevertheless one single numerical measure of quality, it has been proposed to use a weighted harmonic mean of precision and recall, the so-called F_β-measure

$$F_\beta = (1 + \beta^2) \cdot \frac{P \cdot R}{\beta^2 \cdot P + R}. \tag{2.3}$$

We stick to the case $\beta = 1$, i.e. the F_1-measure, because precision and recall are evenly weighted. For the sake of simplicity, we refer to the F_1-measure as F-measure.

Compared to the arithmetic mean, the harmonic mean exaggerates the influence of lower input values and thereby prevents a good F-measure rating when one of the two original measures, precision and recall, is particularly poor.

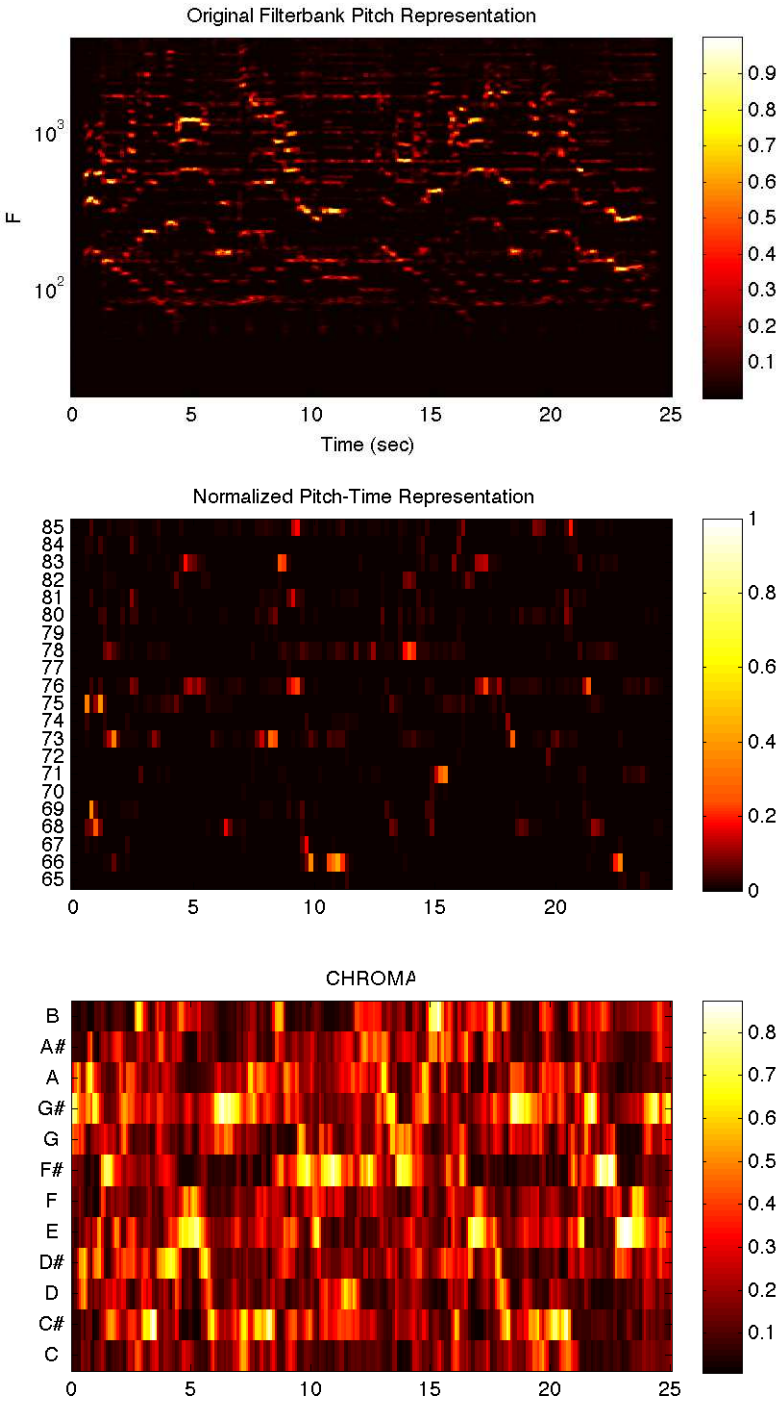

Figure 2.1: Frequency-time, pitch-time and chroma-time representation of "All My Loving" by The Beatles (see also number 1 in Table A.1). **Top:** Frequency-time representation. The rows correspond to frequencies in Hz. Colors correspond to the logarithmic magnitude. **Middle:** Normalized pitch-time representation. The spectrogram is cropped to show only the MIDI pitches 65 to 85 and is rescaled to fit the color coding. The MIDI pitch 69 corresponds to the note A4. **Bottom:** Normalized chroma-time representation. In this representation all the occurrences of a note (also shifted by octaves) are matched to one corresponding row (corresponding pitch classes).

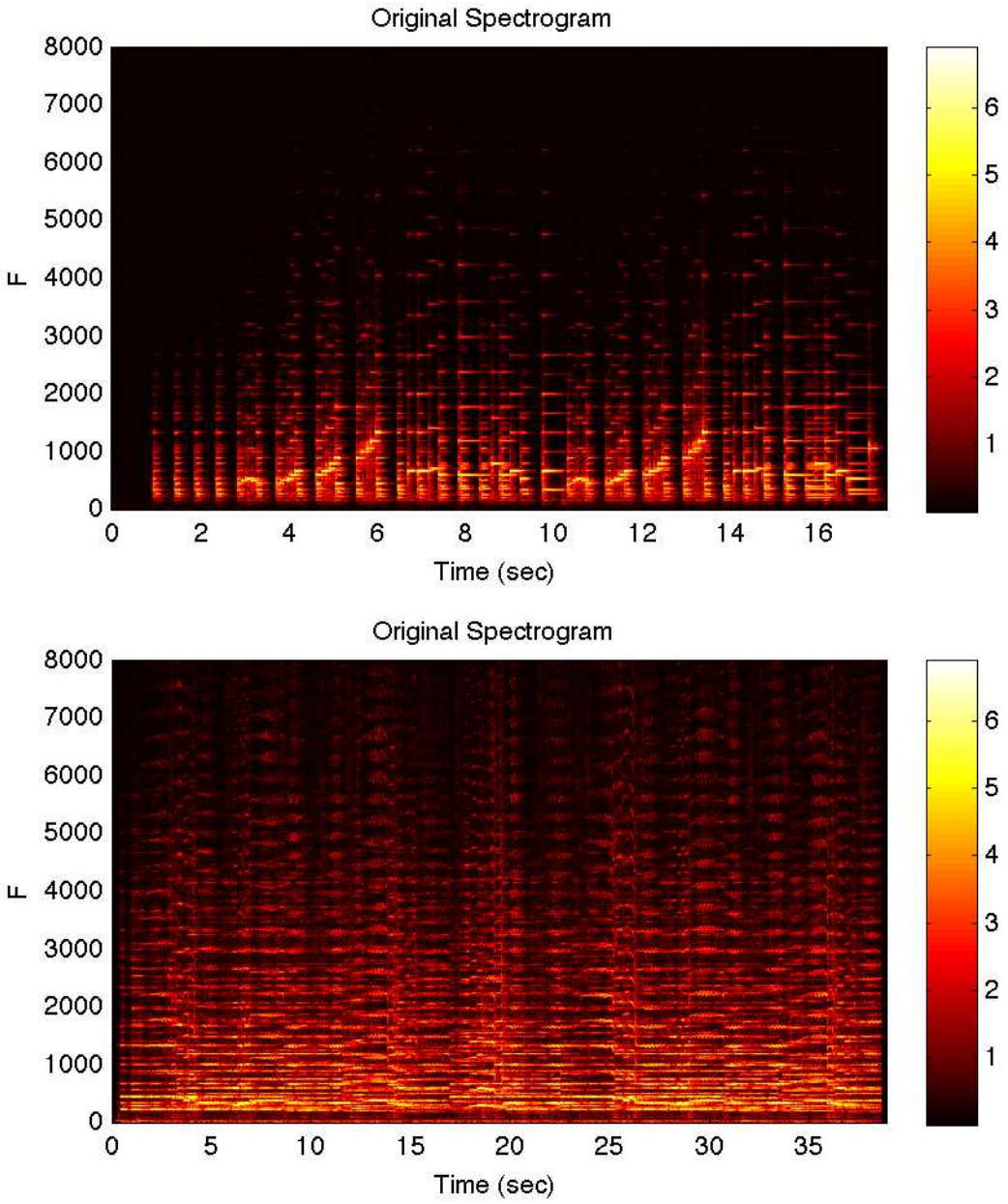

Figure 2.2: Two frequency-time representations. **Top:** Excerpt of the piano etude Op. 100, No. 2 by Friedrich Burgmüller (see also number 1 in Table A.2). Onsets are pretty clear to spot. **Bottom:** Excerpt of the string quartet No. 2 (Notturno) by Alexander Borodin (number 7 in Table A.2). Onsets are difficult to spot.

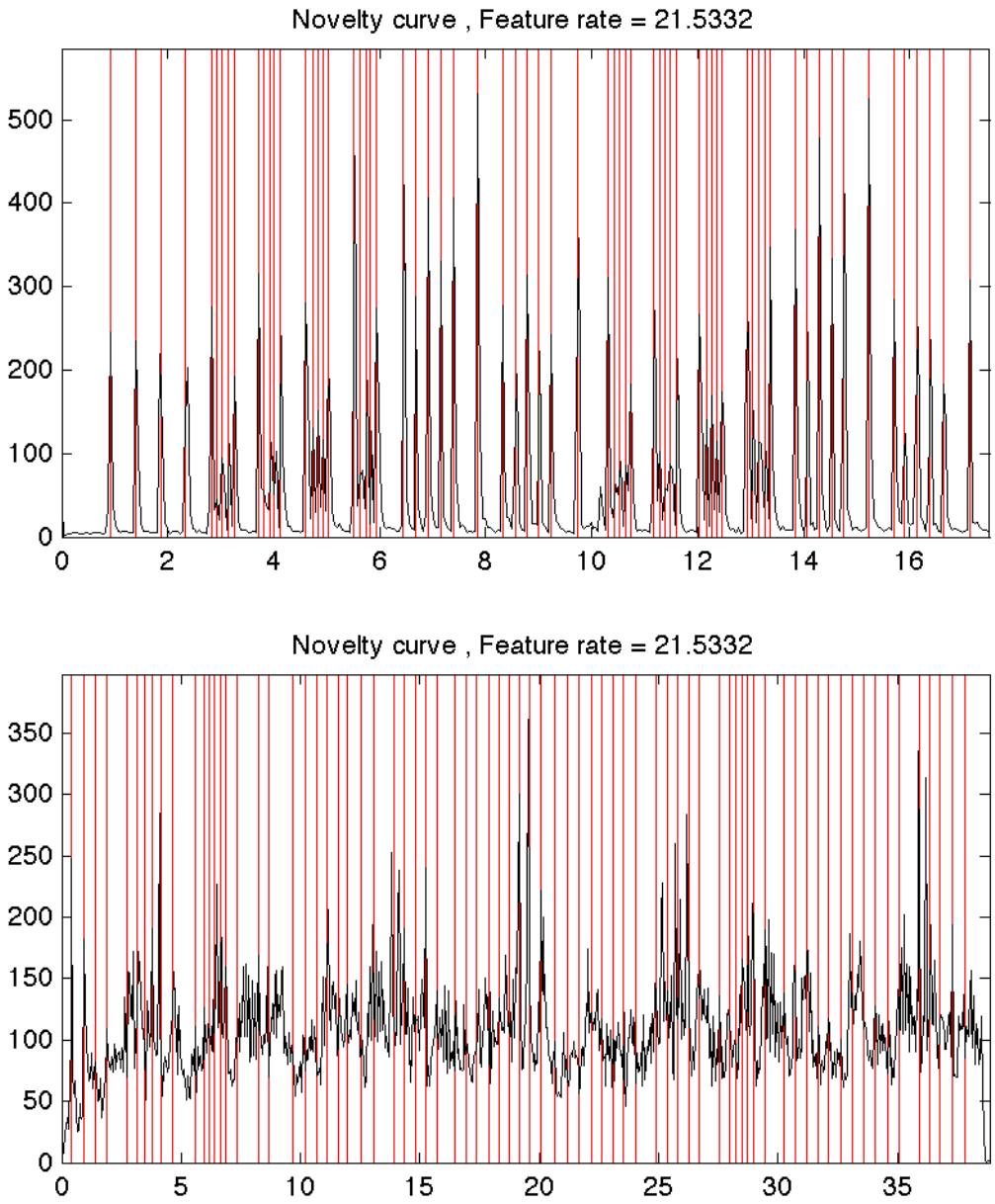

Figure 2.3: Novelty curves corresponding to Figure 2.2. Depicted are the changes in energy as well as the manually annotated ground truth (red lines).

Chapter 3

Diffusion

All science is either physics or stamp collecting.
— Lord Kelvin

Diffusion processes are motivated from physics. Diffusion equilibrates concentration differences by redistributing mass while preserving the total mass. This property is expressed by *Fick's law*

$$j = -D \cdot \nabla u. \tag{3.1}$$

Equation (3.1) means that the concentration gradient ∇u causes a flux j which aims to compensate the gradient. The *diffusion tensor* D – usually a positive definite symmetric matrix – describes the relation between ∇u and j. We differentiate between two cases: *isotropic* and *anisotropic diffusion*. If j and ∇u are parallel, the diffusion is isotropic. In the general anisotropic case, j and ∇u are not parallel.

Since we know that a diffusion process preserves the total mass, we can express this fact in the *continuity equation*

$$\partial_t u = -\text{div } j, \tag{3.2}$$

where t denotes time.

Plugging Fick's law into the continuity equation we get

$$\partial_t u = \text{div}(D \cdot \nabla u). \tag{3.3}$$

Equation (3.3) is called *diffusion equation* and in the case of heat transfer, it is called *heat equation*.

A detailed investigation of diffusion processes can be found in [32].

3.1 The Structure Tensor

The *structure tensor* was introduced by Förstner and Gülch [15] and is a well-known tool for analyzing local orientation. Part of this analysis can be features like corners or edges.

Let us consider a matrix J which results from the tensor product

$$J(\nabla u_\sigma) := \nabla u_\sigma \nabla u_\sigma^\top \tag{3.4}$$

where ∇u_σ is the gradient of u presmoothed by a Gaussian kernel of size σ. ∇u_σ is a vector-valued structure descriptor. The matrix $J(\nabla u_\sigma)$ has an orthonormal basis of eigenvectors v_1 and v_2 with $v_1 \| \nabla u_\sigma$ and $v_2 \perp \nabla u_\sigma$. For a grayscale image, $J(\nabla u_\sigma)$ is a matrix of rank 1 which means that the eigenvalue in the direction of the eigenvector v_2 is 0.

Convolving $J(\nabla u_\sigma)$ component-wise with a Gaussian kernel K_ρ, we obtain the structure tensor

$$J_\rho(\nabla u_\sigma) := K_\rho * (\nabla u_\sigma \nabla u_\sigma^\top) = \begin{pmatrix} K_\rho * (u_x^2) & K_\rho * (u_x u_y) \\ K_\rho * (u_x u_y) & K_\rho * (u_y^2) \end{pmatrix} \quad \text{with } \rho \geq 0. \tag{3.5}$$

The structure tensor J_ρ is symmetric and positive definite. Its orthonormal eigenvectors v_1 and v_2 specify the preferred structure directions within some *integration scale* ρ. The corresponding eigenvalues μ_1 and μ_2 describe the average contrast along the eigendirections and allow for a useful analysis of the local image structure. Thus, the integration scale ρ should reflect the characteristic window size over which the orientation is to be analyzed. Presmoothing in order to obtain ∇u_σ makes the structure tensor insensitive to noise. The parameter σ is called *noise scale*.

Let us assume that $\mu_1 \geq \mu_2 \geq 0$. In this case we observe that v_1 is the orientation with the highest gray level fluctuations and v_2 points us to the preferred local orientation. Furthermore, the eigenvalues tell us useful information for the analysis of the local structure: constant areas are characterized by $\mu_1 = \mu_2 = 0$, straight edges by $\mu_1 \gg \mu_2 = 0$, corners by $\mu_1 \geq \mu_2 \gg 0$ and the expression

$$(\mu_1 - \mu_2)^2 = (j_{11} - j_{22})^2 + 4j_{12}^2$$

becomes large for anisotropic structures, i.e. $(\mu_1 - \mu_2)^2$ is a measure of anisotropy.

3.2 Edge-Enhancing Diffusion

In the setting of *edge-enhancing diffusion* (EED) [32, 31] we want to prevent the diffusion process from diffusing across edges. Let us assume μ_1 and μ_2 with $\mu_1 \geq \mu_2$ be the eigenvalues of the structure tensor J_ρ and v_1 and v_2 are the corresponding orthonormal eigenvectors.

Since the diffusion tensor should reflect the local image structure, it should be chosen such that it has the same set of eigenvectors v_1 and v_2 as the structure tensor J_ρ.

To achieve that we do not diffuse across edges, we reduce the diffusivity λ_1 perpendicular more and more as μ_1 grows. This can be realized by choosing the eigenvalues of the diffusion tensor such that

$$\lambda_1(\mu_1) := g(\mu_1), \tag{3.6}$$

$$\lambda_2 := 1 \tag{3.7}$$

with

$$g(s) := \begin{cases} 1 & \text{if } s \leq 0 \\ \frac{1}{1+s/\lambda^2} & \text{if } s > 0 \end{cases}, \tag{3.8}$$

where $\mu_1 = |\nabla u_\sigma|^2$ and $\lambda > 0$.

The application of edge-enhancing diffusion on a fingerprint image is depicted in Figure 3.1.

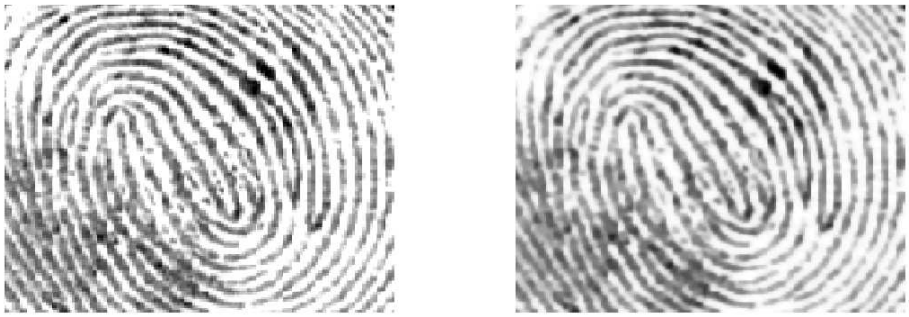

Figure 3.1: Example for an image processed with edge-enhancing diffusion. **Left:** Noisy fingerprint image. **Right:** Same image processed using edge-enhancing diffusion: $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$, $t = 20$.

3.3 Coherence-Enhancing Diffusion

In the context of *coherence-enhancing diffusion* (CED) [32, 33] we want to use the information provided by the structure tensor to design an anisotropic filter that enhances *flow-like structures*. Again, the diffusion tensor should reflect the local image structure.

Let again μ_1 and μ_2 with $\mu_1 \geq \mu_2$ be the eigenvalues of the structure tensor J_ρ and v_1 and v_2 be the corresponding eigenvectors. If we want to enhance coherent structures, we have to smooth preferably along v_2 with a diffusivity of λ_2. This can be achieved by choosing the eigenvalues of the diffusion tensor in the following way:

$$\lambda_1 := \alpha, \tag{3.9}$$

$$\lambda_2 := \begin{cases} \alpha & \text{if } \mu_1 = \mu_2 \\ \alpha + (1-\alpha)\exp\left(\frac{-C}{(\mu_1-\mu_2)^2}\right) & \text{else} \end{cases}, \tag{3.10}$$

where $C > 0$ acts as a threshold parameter and $\alpha \in (0, 1)$ is a small positive parameter. The exponential function and the parameter α are introduced for theoretical reasons: The exponential function ensures the smoothness of the diffusion tensor and the parameter α keeps the diffusion tensor uniformly positive definite. In other words, α acts as a regularization parameter.

The enhancement of a fingerprint image using coherence-enhancing diffusion is shown in Figure 3.2.

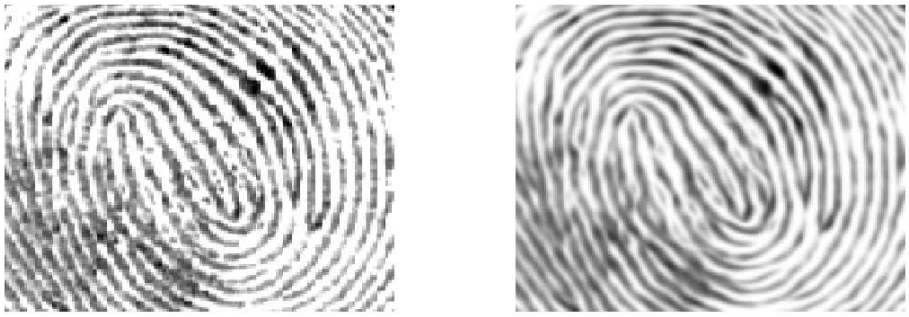

Figure 3.2: Example for an image processed with coherence-enhancing diffusion. **Left:** Noisy fingerprint image. **Right:** Same image processed using coherence-enhancing diffusion: $C = 1$, $\sigma = 0.5$, $\rho = 4$, $\alpha = 0.1$, $\tau = 0.25$, $t = 20$.

3.4 Numerical Method

We briefly mention how to realize a diffusion process algorithmically. The first step is to rewrite the continuity equation (3.3), resulting in

$$\partial_t u = \partial_x \left(D \partial_x u \right) + \partial_y \left(D \partial_y u \right). \tag{3.11}$$

We use a *forward difference* for the time derivative $\partial_t u$

$$\partial_t u \approx \frac{u_{i,j}^{k+1} - u_{i,j}^{k}}{\tau}. \tag{3.12}$$

To compute the structure tensor in each pixel (i, j), we discretize the spatial derivatives $\partial_x u$ and $\partial_y u$ by *central differences*

$$(\partial_x u)_{i,j}^k \approx \frac{u_{i+1,j}^k - u_{i-1,j}^k}{2h_x}, \tag{3.13}$$

$$(\partial_y u)_{i,j}^k \approx \frac{u_{i,j+1}^k - u_{i,j-1}^k}{2h_y}, \tag{3.14}$$

where h_x denotes the spatial grid size in x-direction and h_y the spatial grid size in y-direction (both usually set to 1). The diffusion tensor

$$D_{i,j}^k = \begin{pmatrix} a_{i,j}^k & c_{i,j}^k \\ c_{i,j}^k & b_{i,j}^k \end{pmatrix} \tag{3.15}$$

is computed using the structure tensor, compare Sections 3.2 and 3.3.

Furthermore, we use central differences twice with half the step size for $\partial_x(au_x)$ and $\partial_y(bu_y)$.

$$\begin{aligned} \Big(\partial_x(au_x)\Big)_{i,j}^k &\approx \frac{(au_x)_{i+\frac{1}{2},j}^k - (au_x)_{i-\frac{1}{2},j}^k}{h_x} \\ &\approx \frac{1}{h_x}\left(\frac{a_{i+1,j}^k + a_{i,j}^k}{2} \cdot \frac{u_{i+1,j}^k - u_{i,j}^k}{h_x} \right. \\ &\qquad \left. - \frac{a_{i,j}^k + a_{i-1,j}^k}{2} \cdot \frac{u_{i,j}^k - u_{i-1,j}^k}{h_x} \right), \end{aligned} \tag{3.16}$$

where $(\partial_y(bu_y))_{i,j}^k$ can be computed analogously. Furthermore, we compute the mixed derivatives by

$$u_{xy} = (u_{\xi\xi} - u_{\eta\eta}) \cdot \frac{h_x^2 + h_y^2}{4h_x h_y} \tag{3.17}$$

using the unit vectors

$$\xi = \frac{1}{\sqrt{h_x^2+h_y^2}} \begin{pmatrix} h_x \\ h_y \end{pmatrix}, \tag{3.18}$$

$$\eta = \frac{1}{\sqrt{h_x^2+h_y^2}} \begin{pmatrix} h_x \\ -h_y \end{pmatrix}. \tag{3.19}$$

The discretization of $u_{\xi\xi}$ and $u_{\eta\eta}$ works analogously to the discretization above where the step size is $\sqrt{h_x^2+h_y^2}$ along the diagonal.

The resulting *explicit scheme* reads

$$\begin{aligned} u_{i,j}^{k+1} &= u_{i,j}^k \\ &+ \tau \left(\frac{(a_{i+1,j}^k + a_{i,j}^k)(u_{i+1,j}^k - u_{i,j}^k)}{2h_x^2} - \frac{(a_{i,j}^k + a_{i-1,j}^k)(u_{i,j}^k - u_{i-1,j}^k)}{2h_x^2} \right. \\ &\quad + \frac{(b_{i,j+1}^k + b_{i,j}^k)(u_{i,j+1}^k - u_{i,j}^k)}{2h_y^2} - \frac{(b_{i,j}^k + b_{i,j-1}^k)(u_{i,j}^k - u_{i,j-1}^k)}{2h_y^2} \\ &\quad + \frac{(c_{i+1,j+1}^k + c_{i,j}^k)(u_{i+1,j+1}^k - u_{i,j}^k)}{4h_x h_y} - \frac{(c_{i,j}^k + c_{i-1,j-1}^k)(u_{i,j}^k - u_{i-1,j-1}^k)}{4h_x h_y} \\ &\quad \left. - \frac{(c_{i+1,j-1}^k + c_{i,j}^k)(u_{i+1,j-1}^k - u_{i,j}^k)}{4h_x h_y} + \frac{(c_{i,j}^k + c_{i-1,j+1}^k)(u_{i,j}^k - u_{i-1,j+1}^k)}{4h_x h_y} \right). \end{aligned} \tag{3.20}$$

This scheme is called explicit, since $u_{i,j}^{k+1}$ can be computed explicitly from the known values at the time step k.

Chapter 4

Morphological Filters

Make everything as simple as possible, but not simpler.
— Albert Einstein

Mathematical morphology is one of the most successful classes of image analysis methods. It analyzes the shape of objects in an image.

An interesting property is that morphological filters are invariant under monotone increasing grayscale transformations. It holds that

$$MGf = GMf \tag{4.1}$$

for an image f, where M is any morphological filter and G is any monotonically increasing grayscale transformation. As a consequence, morphological methods do only take into account the *level sets*

$$L_g(f) := \{(x, y) \mid f(x, y) \geq g\} \tag{4.2}$$

of an image f. A detailed discussion of mathematical image analysis can be found in [28].

We now investigate the different building blocks of mathematical morphology.

4.1 Dilation and Erosion

Dilation is one of the basic operations in mathematical morphology. It uses a so-called *structuring element* (also called mask) for probing and expanding the shapes contained in the input image.

The dilation operation on a continuous image $f(x, y)$ using a structuring element SE is defined by

$$(f \oplus SE)(x, y) := \sup\{f(x - x', y - y') \,|(x', y') \in SE\}. \tag{4.3}$$

Erosion is the opposite operation of dilation. It uses the structuring element to shrink the shapes contained in the input image.

Analogously to the dilation operation, we can formulate the erosion of a continuous image $f(x, y)$ using a structuring element SE by

$$(f \ominus SE)(x, y) := \inf\{f(x + x', y + y') \,|(x', y') \in SE\}. \tag{4.4}$$

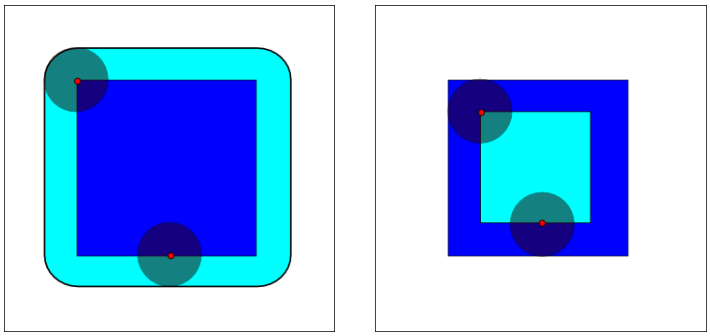

Figure 4.1: Left: The dilation of the dark-blue square by a disk, resulting in the light-blue square with rounded corners. **Right:** The erosion of the dark-blue square by a disk, resulting in the light-blue square. **Source:** Wikipedia [1].

In the discrete case, the maximum is used instead of the supremum and the minimum instead of the infimum.

When applying dilation and erosion often convex structuring elements such as disks or squares are used. Convex structuring elements have the advantage that they are scalable. This means that a dilation/erosion with a structuring element $n \cdot SE$ is equivalent to a dilation/erosion with structuring element SE applied n times.

Figure 4.1 shows how a dilation and an erosion with a disk works.

4.2 Opening and Closing

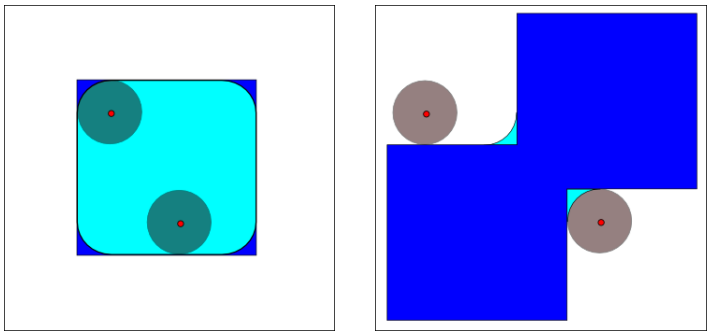

Figure 4.2: Left: The opening of the dark-blue square by a disk, resulting in the light-blue square with round corners. **Right:** The closing of the dark-blue shape (union of two squares) by a disk, resulting in the union of the dark-blue shape and the light-blue areas. **Source:** Wikipedia [1].

Opening and closing are morphological low pass filter. To perform the closing operation, a dilation followed by an erosion is performed. The operation closes small gaps in bright structures and is defined as

$$f \bullet SE := (f \oplus SE) \ominus SE. \tag{4.5}$$

The opening is the opposite operation to closing and is an erosion followed by a dilation. Mathematically, it is defined as

$$f \circ SE := (f \ominus SE) \oplus SE. \tag{4.6}$$

It closes small gaps in dark structures. A visualization of these operations is given in Figure 4.2.

4.3 Nonflat Morphology

In the previous sections, we saw examples of morphological operations with a disk which is a flat structuring element. In the context of grayscale images also nonflat structuring elements can be used. Motivated by van den Boomgaard [30], we present modified versions of dilation and erosion that use a parabola as structuring element (to be more precise a *structuring function*) where we stick to the discrete case.

The *parabolic dilation* reads as

$$f_d(x,y) := \max_{i,j} \Big(f(i,j) - \big(s_x \cdot (x-i)^2 + s_y \cdot (y-j)^2 \big) \Big) \tag{4.7}$$

and the *parabolic erosion* as

$$f_e(x,y) := \min_{i,j} \Big(f(i,j) + \big(s_x \cdot (x-i)^2 + s_y \cdot (y-j)^2 \big) \Big), \tag{4.8}$$

where $(x,y),(i,j) \in I$ and s_x and s_y are scaling parameters.

Using a parabolic structuring element introduces some challenges. Nonflat morphological methods do not operate on level sets as given in (4.2). In other words, equation (4.1) is not satisfied anymore. Another practical difficulty with nonflat morphological operations arises with spatial discretization. While parabolic dilation and erosion in their continuous formulation retain the property that dilating or eroding n times with the same structuring element is equivalent to dilating or eroding once with a structuring element rescaled to n times its horizontal size, the same does not even approximately hold when parabolic dilation and erosion are carried out on a spatial grid. We will have to be considerate of this circumstance when applying nonflat morphology for skeletonization in Section 6.2.

4.4 Skeletonization

Skeletonization was already investigated in 1968 by Calabi and Hartnett [9] and is still a topic of active research (e.g. Kimmel et al. [18]) where people are also interested in fast algorithms [35]. Skeletons are frequently used as shape descriptors, e.g. in character recognition.

We can think of the skeletonization as follows. Imagine some arbitrary shape. We set the boundary of this shape afire. Then, the fire moves from the boundary into the shape where the *Huygens wavefronts* of the boundary points meet at the *skeleton point*. In other words, the

skeleton point is located at the intersection of the normals to the two boundary points and each skeleton point corresponds to at least two boundary points. The distance between the skeleton point to both boundary points is equal. Figure 4.3 shows two shapes and their corresponding skeletons given as a thick black line.

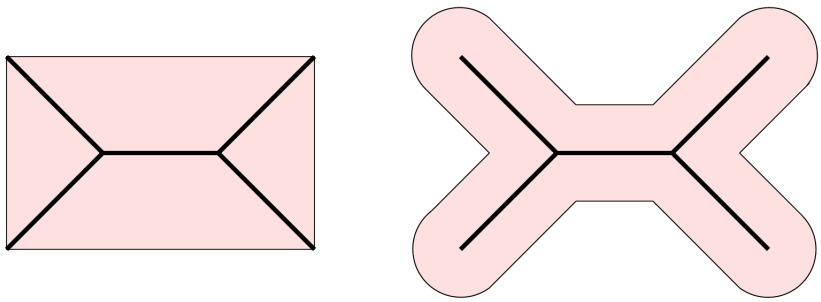

Figure 4.3: Two shapes and their corresponding skeletons. Each skeleton is given as a thick black line. **Source:** [5].

After having some intuition, let us formulate the skeletonization more mathematically. The skeleton of a set can be expressed by erosions and openings. We have

$$S(f) := \bigcup_{k=0}^{K} \Big\{ (f \ominus k \cdot SE) - \big[(f \ominus k \cdot SE) \circ SE \big] \Big\}, \tag{4.9}$$

where SE is the structuring element and K is the last iterative step before f erodes to an empty set. Please note that equation (4.9) works on binary images only.

Chapter 5

Non-Local Means (NL-Means)

Science is always wrong. It never solves a problem without creating ten more.
— George Bernard Shaw

NL-means was proposed by Buades et al. [8]. The algorithm tries to replace a noisy pixel with a weighted average of other pixels with similar neighborhoods. The basic idea is that images contain self-similarities and many structures show up more than once in the image where we usually assume that noise in similar neighborhoods is uncorrelated. Thus, taking the average of similar neighborhoods yields a version of the image with less noise.

5.1 Basic NL-Means

Given a discrete noisy image $u = \{u(\boldsymbol{i}) \mid \boldsymbol{i} \in I\}$, the estimated value $NL[u](\boldsymbol{i})$ for a pixel $\boldsymbol{i} = (i, j)$ is computed as a weighted average of all the pixels in the image:

$$NL[u](\boldsymbol{i}) := \sum_{\boldsymbol{j} \in I} w(\boldsymbol{i},\boldsymbol{j}) u(\boldsymbol{j}), \tag{5.1}$$

where $\boldsymbol{j} = (i', j') \in I$ and the usual conditions $0 \leq w(\boldsymbol{i},\boldsymbol{j}) \leq 1$ and $\sum_{\boldsymbol{j}} w(\boldsymbol{i},\boldsymbol{j}) = 1$ are satisfied.

The similarity between two pixels $\boldsymbol{i}$ and $\boldsymbol{j}$ depends on the similarity of the intensity gray level vectors $u(\mathcal{N}_{\boldsymbol{i}})$ and $u(\mathcal{N}_{\boldsymbol{j}})$, where $\mathcal{N}_{\boldsymbol{i}}$ and $\mathcal{N}_{\boldsymbol{j}}$ denote the neighborhood of fixed size centered at pixel $\boldsymbol{i}$ and $\boldsymbol{j}$, respectively. Pixels with a similar gray level neighborhood to $u(\mathcal{N}_{\boldsymbol{i}})$ have larger weights in the average. The weights are defined as follows:

$$w(\boldsymbol{i},\boldsymbol{j}) := \frac{1}{Z(\boldsymbol{i})} \exp\left(-\frac{\|u(\mathcal{N}_{\boldsymbol{i}}) - u(\mathcal{N}_{\boldsymbol{j}})\|^2_{2,a}}{\lambda^2}\right), \tag{5.2}$$

where $Z(\boldsymbol{i})$ is a normalization constant

$$Z(\boldsymbol{i}) := \sum_{\boldsymbol{j}} \exp\left(-\frac{\|u(\mathcal{N}_{\boldsymbol{i}}) - u(\mathcal{N}_{\boldsymbol{j}})\|^2_{2,a}}{\lambda^2}\right) \tag{5.3}$$

and λ is the decay parameter of the exponential function. The expression $\|u(\mathcal{N}_i) - u(\mathcal{N}_j)\|^2_{2,a}$ is an Euclidean distance, where $a > 0$ is the standard deviation of the Gaussian kernel. In other words, the parameter λ in equation (5.2) controls the weighting of the distances.

This weighting function penalizes structures which have less similarity in the geometrical configuration of their neighborhood, see Figure 5.1.

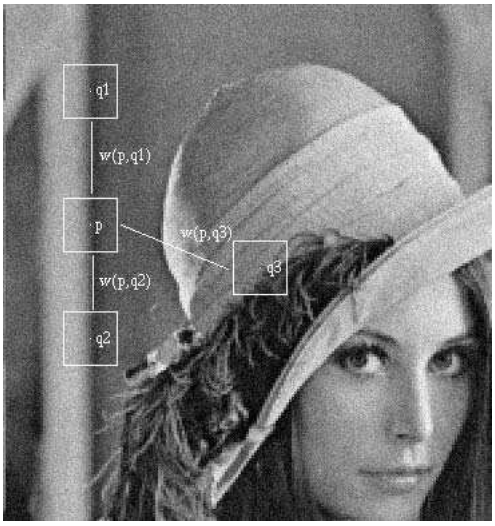

Figure 5.1: Scheme of NL-means strategy. Similar pixel neighborhoods give a large weight, $w(\boldsymbol{p}, \boldsymbol{q}_1)$ and $w(\boldsymbol{p}, \boldsymbol{q}_2)$, while much different neighborhoods give a small weight, $w(\boldsymbol{p}, \boldsymbol{q}_3)$. **Authors:** A. Buades, B. Coll and J.-M. Morel [8].

5.2 Modifications and Refinements to NL-means

The main drawback of the basic NL-means algorithm is its high complexity, a fact that was also noted by Buades et al. in [8]. To address this issue, many ideas were proposed to reduce the complexity of the computation and thus also to reduce the runtime of the algorithm. In the following, we introduce some of them.

5.2.1 Restricting the Search Space of NL-Means

In its very basic form, NL-means compares every pixel of the image with the pixel that should be replaced to find a possible denoising candidate. This is computationally very expensive and thus not very practical. To overcome this problem, Buades et al. [8] introduced a search window (a neighborhood) which reduces the search space dramatically.

Let us assume that N^2 is the number of pixels of the image. Let us further assume that the similarity window has the size $W \times W$ and the search window has size $S \times S$. In its basic form, the algorithm has complexity $W^2 \times N^2 \times N^2$. With the introduction of the search window, the algorithm has complexity $W^2 \times S^2 \times N^2$ where $S^2 \ll N^2$.

5.2.2 Neighborhood Classification to Speedup NL-Means

Let us now explore how neighborhood classification can help to lower the algorithmic complexity of NL-means. We look at a neighborhood classification that works with two components. The first one is based on *average neighborhood gray values* and the second one is based on *gradient directions*. We follow here the work proposed by Mahmoudi and Sapiro [23].

Intuitively, similar neighborhoods should have similar average gray values. Let $\overline{u}(\boldsymbol{i})$ and $\overline{u}(\boldsymbol{j})$ be the average gray value in the neighborhoods of pixels $\boldsymbol{i}$ and $\boldsymbol{j}$, respectively. Let furthermore $\eta_1 < 1$ and $\eta_2 > 1$ be two constants close to 1. The weight $w(\boldsymbol{i},\boldsymbol{j})$ has a non-zero value, i.e. the neighborhood is only considered, if

$$\eta_1 < \frac{\overline{u}(\boldsymbol{i})}{\overline{u}(\boldsymbol{j})} < \eta_2 \tag{5.4}$$

holds.

A second method to approximate the similarity between two neighborhoods is their *average gradient orientation*. If $\nabla u(\boldsymbol{i}) = (u_x(\boldsymbol{i}), u_y(\boldsymbol{i}))$ is the image gradient then the average gradient orientation around pixel $\boldsymbol{i}$ is defined as follows:

$$\overline{\nabla u}(\boldsymbol{i}) = (\overline{u_x}(\boldsymbol{i}), \overline{u_y}(\boldsymbol{i}))\,, \tag{5.5}$$

where $\overline{u_x}(\boldsymbol{i})$ and $\overline{u_y}(\boldsymbol{i})$ are the average x and y derivatives in the neighborhood of pixel $\boldsymbol{i}$. Having this, we can formulate the difference of the average gradient orientations at pixels $\boldsymbol{i}$ and $\boldsymbol{j}$:

$$\theta(\boldsymbol{i},\boldsymbol{j}) = \angle(\overline{\nabla u}(\boldsymbol{i}), \overline{\nabla u}(\boldsymbol{j})). \tag{5.6}$$

To find a threshold above which θ is considered as outlier, Mahmoudi et al. propose to use robust statistics, such as

$$\sigma_\theta = 1.4826 \cdot \text{median}_{I\times I}\Big[\big|\,|\theta(\boldsymbol{i},\boldsymbol{j})| - \text{median}_{I\times I}(|\theta(\boldsymbol{i},\boldsymbol{j})|)\big|\Big]. \tag{5.7}$$

To compute a threshold for the gradient, we do a similar computation:

$$\sigma_\nabla = 1.4826 \cdot \text{median}_{I\times I}\Big[\big|\,|\nabla u|^2 - \text{median}_{I\times I}(|\nabla u|^2)\big|\Big]. \tag{5.8}$$

Plugging everything together, we end up with the following weighting

$$w(\boldsymbol{i},\boldsymbol{j}) := \begin{cases} \frac{1}{Z(\boldsymbol{i})} \exp\left(-\frac{\|u(\mathcal{N}_\boldsymbol{i}) - u(\mathcal{N}_\boldsymbol{j})\|^2_{2,a}}{\lambda^2}\right), & \big[(\|\nabla u(\boldsymbol{i})\| < \sigma_\nabla) \\ & \text{or } (\|\nabla u(\boldsymbol{j})\| < \sigma_\nabla) \\ & \text{or } (|\theta(\boldsymbol{i},\boldsymbol{j})| < \sigma_\theta)\big] \\ & \text{and } (\eta_1 < \frac{\overline{u}(\boldsymbol{i})}{\overline{u}(\boldsymbol{j})} < \eta_2) \\ 0, & \text{otherwise} \end{cases}. \tag{5.9}$$

5.2.3 Locally Adaptive Weighting for NL-Means

Since the parameter λ in equation (5.2) controls the weighting of the distances, it is the most important and crucial parameter of NL-means: Choosing it too low can lead to a noisy result, choosing it too high can blur fine structures. Brox and Cremers [7] showed that there is in general no global value for λ that is well suited for the whole image. In the following we look at an idea proposed by Zimmer et al. [34].

Instead of using a fixed λ, a *locally adaptive function* $\Lambda(\lambda, B)$ is introduced. The authors' first proposal is to use the standard deviation s_B of a block B. The standard deviation depends on the block structure and on the amount of noise contained in B. Sharp structures usually mean that there are few similarities, since sharp structures lead to a high variance. Thus, we have to smooth more there compared to blocks where the variance is low. To avoid a too high influence on the smoothing, Zimmer et al. propose to use a sub-linear function Ψ. Finally, we end up with

$$\Lambda(\lambda, B) := \lambda \cdot \Psi(s_B), \tag{5.10}$$

where $\Psi(s_B) = \sqrt{s_B}$. Since this locally adaptive function replaces λ in equation (5.2), we refer to it by a formulation like *adaptively chosen (distance weight)* λ or *adaptive (distance weight)* λ.

Chapter 6

Modifications and Refinements

In the beginner's mind there are many possibilities. In the expert's mind there are few.
— Shunryu Suzuki

In this chapter, we want to investigate modifications and refinements to the algorithms introduced in the chapters before. The modifications address drawbacks concerning runtime or provide refinements that allow a better processing of spectrograms in the context of chord recognition or onset detection.

6.1 Diffusion

The diffusion tensor D is the tool to control a diffusion process. We make use of the structure tensor J to compute D

$$D = \begin{pmatrix} a & c \\ c & b \end{pmatrix}, \tag{6.1}$$

where the eigenvectors of D are identical to those of J while the eigenvalues are recomputed to fit equations (3.6)–(3.8) for edge-enhancing diffusion or equations (3.9),(3.10) for coherence-enhancing diffusion, respectively. Since we are interested in horizontal structures in the context of chord recognition and in vertical structures in the context of onset detection, we modify the diffusion tensor to prefer diffusion in a certain direction.

Multiplying the diffusion tensor with a matrix of the form

$$\begin{pmatrix} d_\alpha & 0 \\ 0 & d_\beta \end{pmatrix}$$

from both sides introduces anisotropy controlled by the choice of d_α and d_β. The *modified diffusion tensor* looks as follows:

$$D' = \begin{pmatrix} d_\alpha & 0 \\ 0 & d_\beta \end{pmatrix} \begin{pmatrix} a & c \\ c & b \end{pmatrix} \begin{pmatrix} d_\alpha & 0 \\ 0 & d_\beta \end{pmatrix} = \begin{pmatrix} d_\alpha^2 a & d_\alpha d_\beta c \\ d_\alpha d_\beta c & d_\beta^2 b \end{pmatrix}. \tag{6.2}$$

The weightings d_α and d_β can be used to enhance or reduce the diffusion in one direction – i.e. choosing a value $d_\alpha, d_\beta \geq 1$ or a value $0 \leq d_\alpha, d_\beta < 1$, respectively. In this thesis, only the values 0 and 1 are used to switch the diffusion process in a certain direction on or off.

6.2 Morphology

In Section 4.3, we stated that for the discrete version of parabolic dilation and parabolic erosion the scalability property is not guaranteed anymore. We now present an approach to overcome this problem.

To get rid of the violation of the scalability property, we modify the scaling parameters s_x and s_y from equations (4.7) and (4.8) in every iteration step. We can do this by introducing scaling parameters s'_x and s'_y such that it holds

$$s'_x = \frac{1}{2^k} \cdot s_x, \tag{6.3}$$

$$s'_y = \frac{1}{2^k} \cdot s_y, \tag{6.4}$$

where $k = 1, \ldots, n$ is the current iteration and n is the total number of iterations.

By means of parabolic dilation and parabolic erosion as given in equations (4.7) and (4.8), respectively, we can think of a new skeletonization approach: *parabolic skeletonization*.

Let SE_k correspond to $(s'_x, s'_y) = (\frac{1}{2^k} \cdot s_x, \frac{1}{2^k} \cdot s_y)$. Then the parabolic skeletonization reads as

$$S(f) = \sum_{k=0}^{n} \Big[f \ominus SE_k - f \circ SE_{k+1} \Big]_+, \tag{6.5}$$

where we only sum up the positive contributions.

Note that this is not an exact generalization of the skeletonization algorithm (4.9) for binary images. The idea behind both algorithms is that in the k-th step (i.e. k-th summand) those parts of the eroded signal $f \ominus SE_k$ should be added to the skeleton which would be destroyed in the next erosion step $f \ominus SE_{k+1}$ "irreversibly", i.e. cannot be restored by dilation. In (4.9) these parts are detected by dilating back the signal $f \ominus SE_{k+1}$ to the level of the k-th erosion and comparing with $f \ominus SE_k$: the subtrahend of (4.9) can as well be written as $(f \ominus (k+1)SE) \oplus SE$. However, as we mentioned in Section 4.3, iterated application of discretized parabolic dilations or erosions with the structuring function SE does not approximate a discretized parabolic dilation or erosion with a scaled structuring function. This poses an obstacle to direct translation of (4.9).

Our solution in (6.5) consists in dilating back $f \ominus SE_{k+1}$ in one step to the level of the initial signal f and contributing to the skeleton those parts of $f \ominus SE_k$ which are *not even then* restored. As $f \circ SE_{k+1}$ is not completely below $f \ominus SE_k$, the restriction to the positive part is necessary. As a consequence, (6.5) applied in a binary morphological setting would lose substantial contributions as compared to (4.9). Experiments show, however, that in conjunction with parabolic structuring functions this approach retains enough signal contributions to create a reasonable, moderately thinned version of image structures.

Furthermore, we introduce an upper bound to keep the runtime of the algorithm short. The algorithm stops at n such that

$$\max\left(f \ominus SE_n - f \circ SE_{n+1}\right) \leq \text{threshold} \tag{6.6}$$

holds. In other words the algorithm stops if another step of the parabolic skeletonization as described in equation (6.5) would not give a much better result compared to the higher runtime.

6.3 NL-Means

In Section 5.2, we have seen some refinements that can be made to improve the NL-means algorithm. Now, we introduce another modification that addresses special needs of chord recognition and onset detection.

So far, we assumed for NL-means a quadratic search and a quadratic similarity window. When it comes to applications like chord recognition or onset detection where we are interested in horizontal or vertical structures, respectively, a quadratic window is not an optimal choice. The logical consequence is to change the windows in such a way that we can have rectangular ones. Analogously to the notation in 5.2.1, we can introduce a similarity window of size $W_x \times W_y$ and a search window of size $S_x \times S_y$. Naturally, the algorithm can also deal with input images that are rectangular, i.e. images of size $N_x \times N_y$.

As already stated, NL-means in its basic form is computationally very expensive and we presented different approaches to address this drawback. Considering the change of the windows as proposed in this section, the complexity of the algorithm is now $(W_x \times W_y) \times (S_x \times S_y) \times (N_x \times N_y)$ which might also lead to a dramatically reduced runtime depending on the window sizes.

Chapter 7

Experiments in Chord Recognition

Celebrate any progress. Don't wait to get perfect.
— Ann McGee Cooper

In this chapter, we will probe the various classes of image processing tools discussed in the preceding chapters in the context of the chord recognition problem. Remember that emphasis is laid here on horizontal structures in the spectrograms; accordingly, the algorithmic variants with horizontal directional preference will receive special attention.

Before entering the experiments, let us say a few words on the experimental setup. For all the computation, MATLAB 7.8.0 (R2009a) was executed on a 2.66 GHz Apple iMac with 4 GB of RAM running Mac OS X 10.6. The largest part of the code was implemented using the MATLAB programming language where in the case of CED and EED the C code provided by the MIA group was used and slightly modified to compile as a MEX file that allows C code to run within the MATLAB environment [2].

The testfiles constitute excerpts of the songs referenced by their title and were annotated by Mauch et al. [24]. The name and the length of the excerpt of each testfile is listed in Table A.1. As a quality measure, precision, recall and F-measure as introduced in Section 2.3 are used.

7.1 Diffusion

In this section, we show how diffusion algorithms perform on the spectrograms for the task of chord recognition.

In the first experiment, we look at the influence of using the diffusion tensor as introduced in Section 6.1 with and without the preference for horizontal structures for a fixed number of iterations. Secondly, we investigate the behavior of the algorithm for different numbers of iterations where d_α is set to 1 and d_β is set to 0, i.e. we prefer a horizontal diffusion.

7.1.1 Edge-Enhancing Diffusion

Horizontal vs. no directional preference. Figure 7.1 shows the spectrogram of "Let It Be" by The Beatles. The top spectrogram depicts the unprocessed data. In the bottom row, the comparison of an edge-enhancing diffusion without and with directional preference for horizontal structures is given. The bottom left spectrogram clearly shows that the diffusion happened in every direction which results also in a vertical blurring of the structures, i.e. the frequencies. In comparison to that, the version with directional preference turned on gives a spectrogram where the diffusion process only acts in horizontal direction which means that it does not blur other frequencies.

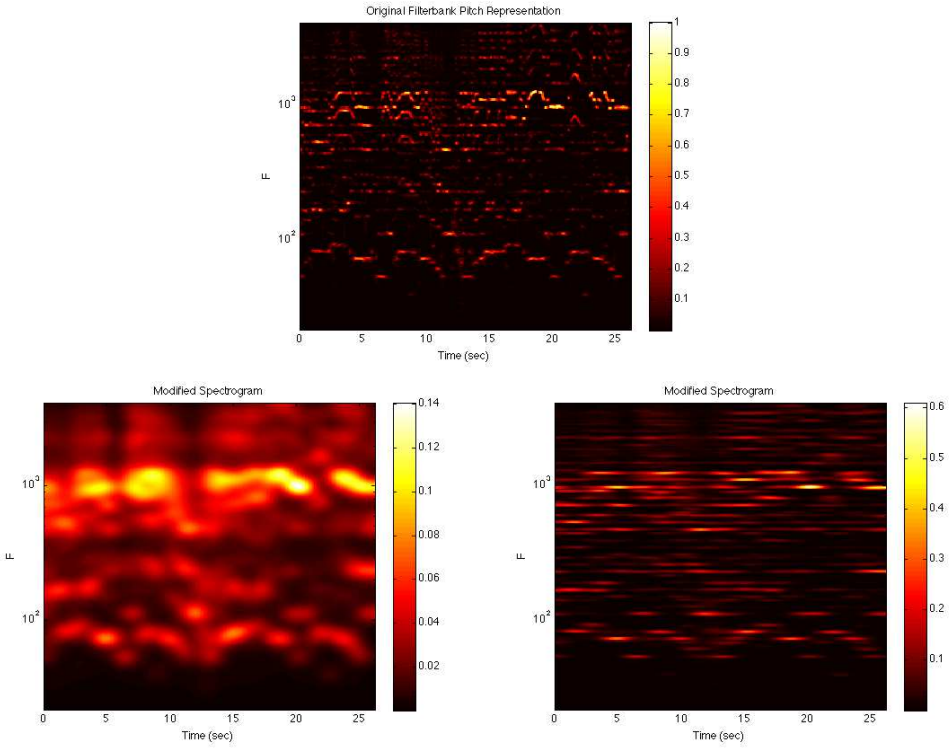

Figure 7.1: Spectrograms of "Let It Be" performed by The Beatles (number 10 in Table A.1). **Top:** Original spectrogram. **Bottom left:** Spectrogram after diffusion with EED and $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$, $t = 100$ where no direction is preferred. **Bottom right:** Spectrogram after diffusion with the same setting, but with horizontal preference.

Table 7.1 also confirms the intuition that a diffusion process that diffuses in every direction does not help in the context of chord recognition. In fact, the structures are blurred in such an intense way that the processed spectrogram is much less suited for feature extraction than the original, unprocessed spectrogram. Furthermore, the information which frequency contributes at which time is completely blurred away. This stands in contrast to the results obtained by the same computation but with preference for horizontal structures turned on. Here we can clearly distinguish the different frequencies involved in this music excerpt.

Table 7.1: Comparison of precision, recall and F-measure for EED with and without directional preference. Edge-enhancing diffusion was performed with $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$ and $t = 100$. The values correspond to the processing depicted in Figure 7.1.

Setting	Precision	Recall	F-Measure
Original	0.496	0.496	0.496
No preference	0.021	0.021	0.021
Horizontal preference	0.585	0.585	0.585

Increasing the number of iterations. In addition to that, Figure 7.2 shows spectrograms processed by edge-enhancing diffusion with directional preference for horizontal structures where the impact of different numbers of iterations is compared. Obviously, the more iterations are performed the stronger the smearing in time. In other words, long diffusion times introduce an error since frequencies appear where they are actually not located. Table 7.2 shows that the higher the number of iterations the lower the F-measure of the result. Let us take as an example the peak located at a frequency slightly below 10^2 that extends in time from shortly before 5 to about 6 seconds. We see that this frequency contributes more and more beyond the original boundaries of this time frame the higher t gets. For $t = 500$ we can observe that a contribution is present almost at every time in this example.

Please also note that for this particular testfile the values for precision and recall given in tables 7.1 and 7.2 are the same – also resulting in the same value for the F-measure, compare equation (2.3) – which does not hold in general.

Table 7.2: Comparison of precision, recall and F-measure for EED with different numbers of iterations and directional preference for horizontal structures where $\lambda = 1$, $\sigma = 0.5$ and $\tau = 0.25$. The values correspond to the processing depicted in Figure 7.2.

Iterations	Precision	Recall	F-Measure
0	0.496	0.496	0.496
50	0.634	0.634	0.634
100	0.585	0.585	0.585
500	0.475	0.475	0.475

7.1.2 Coherence-Enhancing Diffusion

Horizontal vs. no directional preference. Turning to coherence-enhancing diffusion, we start by investigating the impact of steering the diffusion only in horizontal direction in comparison to a diffusion process that diffuses in every direction like we did in the previous section. Figure 7.3 depicts the corresponding spectrograms for “All You Need Is Love” by The Beatles. On the bottom left spectrogram, we can see that unbiased anisotropic diffusion

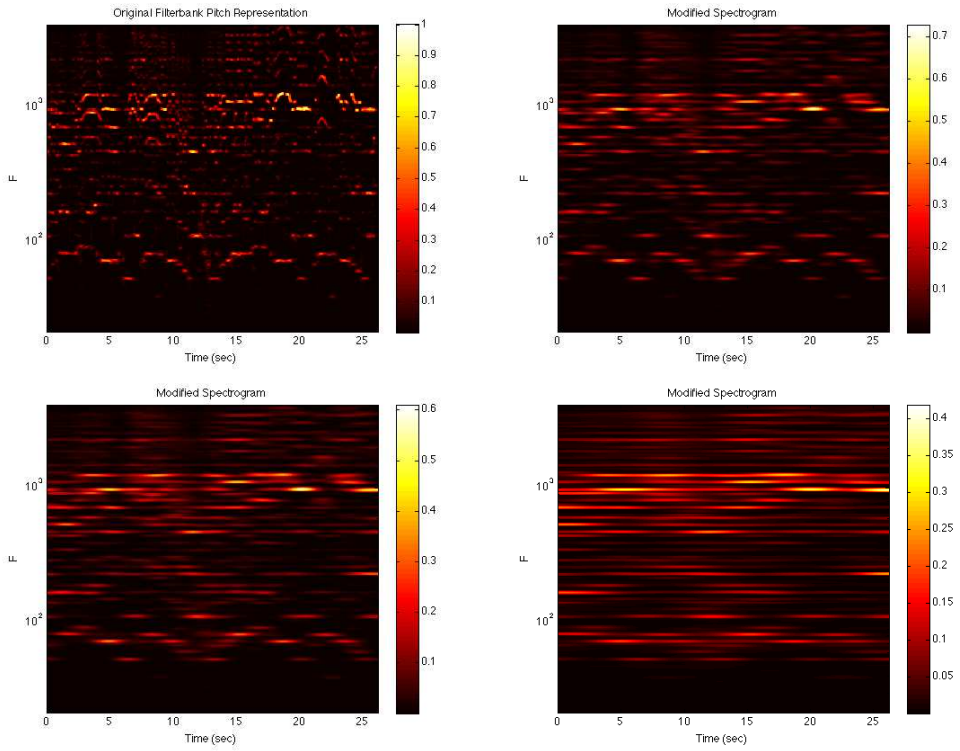

Figure 7.2: Spectrograms of "Let It Be" performed by The Beatles (number 10 in Table A.1). All diffusion processes are performed with a preference for horizontal structures. **Top left:** Original spectrogram. **Top right:** Spectrogram after diffusion with EED and $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$ and $t = 50$. **Bottom left:** $t = 100$. **Bottom right:** $t = 500$.

heavily blurs the structures in the spectrogram. It seems that in the top half of the spectrogram every frequency contributes at every time. This result is far away from being useful which is also affirmed by the quality measure, compare Table 7.3. Again, the change of the diffusion tensor to only allow horizontal diffusion helps to get rid off this instance (Figure 7.3, bottom right). Despite the presence of blurring in the top half of the spectrogram, we may see that the frequencies in the bottom of the spectrogram are well preserved by this modified diffusion.

Increasing the number of iterations. Analyzing Figure 7.4 and the corresponding Table 7.4 we notice something counterintuitive. The more the bright, yellowish frequencies in the bottom half of the spectrogram are blurred, the more enhanced is the result according to the quality measure. We would expect a poor outcome for high iteration numbers, i.e. an higher error with the increase of the number of iterations and thus smaller values for precision, recall and F-measure, but receive the opposite. The reason is the following. Diffusion in horizontal direction introduces the frequencies in the bottom of the spectrogram more and more at every point in time the more iterations are performed. This results in the existence of strong contributions of these frequencies where actual occurrences should be located while the contributions that are spuriously introduced are small securing the result a high quality

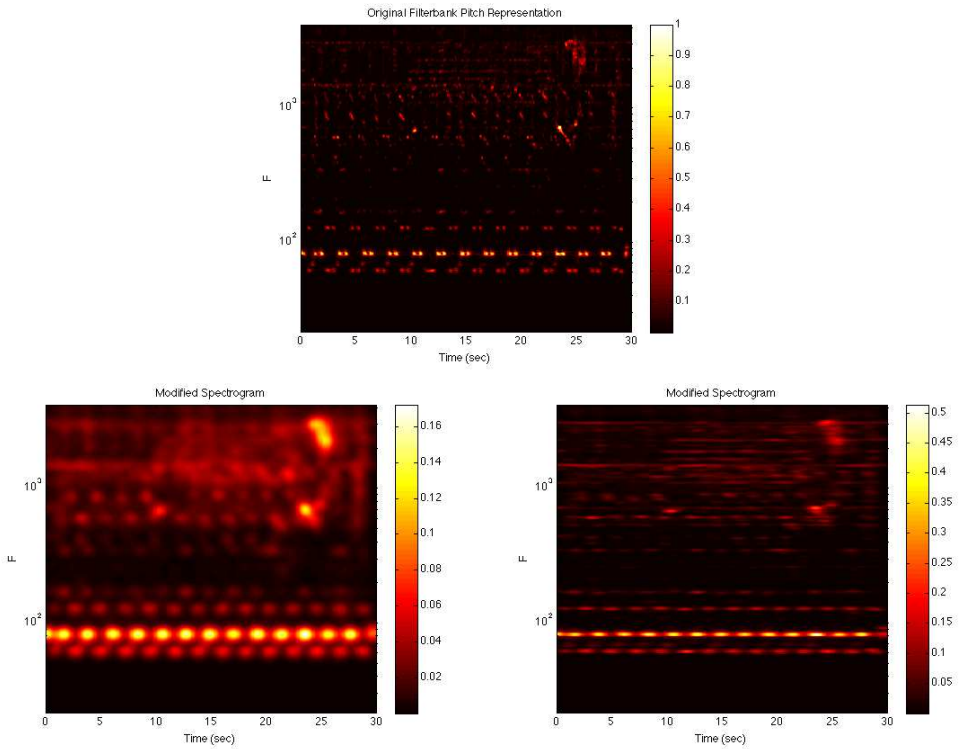

Figure 7.3: Spectrograms of "All You Need Is Love" performed by The Beatles (number 2 in Table A.1). **Top:** Original spectrogram. **Bottom left:** Spectrogram after diffusion with CED and $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 0.1$, $\tau = 0.25$, $t = 500$ where no direction is preferred. **Bottom right:** Spectrogram after diffusion with the same setting, but with preference for horizontal direction.

rating according to the precision, recall and F-measure criteria. This demonstrates that these criteria are not always suitable as measures of quality, and their output should be taken with some caution.

Table 7.3: Comparison of precision, recall and F-measure for CED with and without directional preference where $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 0.1$, $\tau = 0.25$ and $t = 500$. The values correspond to the processing depicted in Figure 7.3.

Setting	Precision	Recall	F-Measure
Original	0.412	0.416	0.414
No preference	0.000	0.000	0.000
Horizontal preference	0.775	0.783	0.779

Table 7.4: Comparison of precision, recall and F-measure for CED ($C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 0.1$ and $\tau = 0.25$) with horizontal preference and different numbers of iterations. The values correspond to the processing depicted in Figure 7.4.

Iterations	Precision	Recall	F-Measure
0	0.412	0.416	0.414
100	0.569	0.575	0.572
500	0.775	0.783	0.779
1000	0.902	0.910	0.906

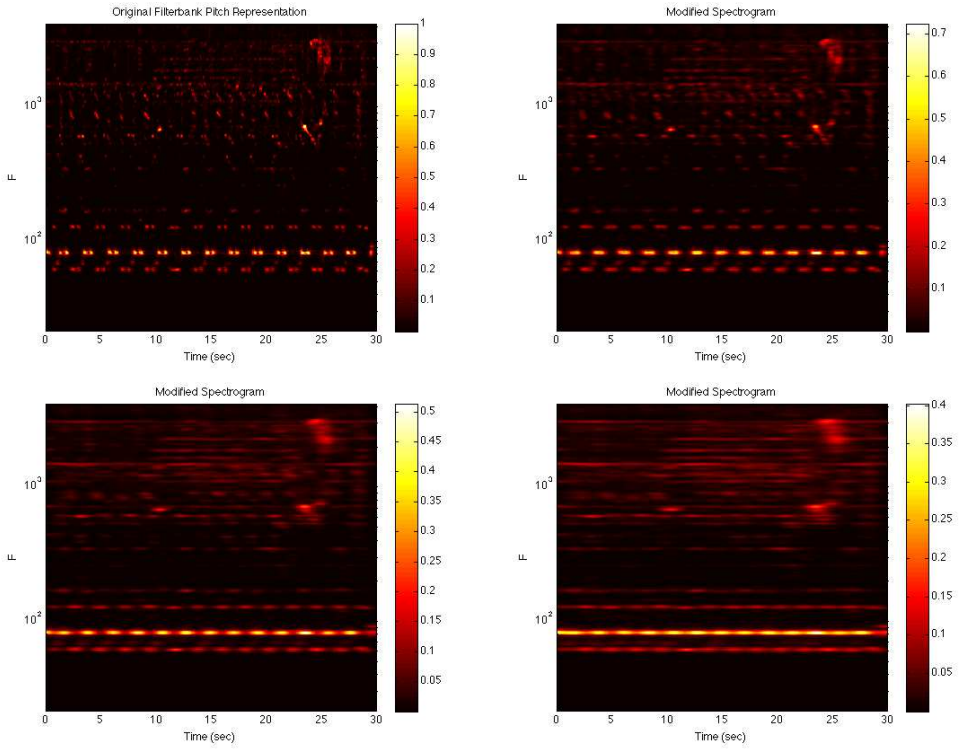

Figure 7.4: Spectrograms of "All You Need Is Love" performed by The Beatles (number 2 in Table A.1). All diffusion processes are performed with a preference for horizontal structures. **Top left:** Original spectrogram. **Top right:** Spectrogram after diffusion with CED and $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 0.1$, $\tau = 0.25$, $t = 100$. **Bottom left:** $t = 500$. **Bottom right:** $t = 1000$.

7.1.3 Comparison of EED and CED

We have seen that EED and CED can perform similarly well in the setting of chord recognition. We conclude this section by comparing edge-enhancing diffusion and coherence-enhancing diffusion head to head with similar parameter settings. Please note that the parameter α in CED is some sort of damping parameter for the diffusion process. If we scale α down by a factor s, we have to perform $t \cdot s$ iterations to get the same result. For the following comparison, we set $\alpha = 1$ to ensure that both EED and CED run with the same number of iterations.

Visually, no difference between the spectrograms on the left and on the right of Figure 7.5 can be identified. Also the quality measurements given in Table 7.5 do not disclose any differences. The natural question is why is this the case? The answer is simple. Coherence-enhancing diffusion enhances flow-like (coherent) structures. In this context, these flow-like structures have the same orientation as the edges. Thus, edge-enhancing diffusion and coherence-enhancing diffusion compute a similar structure tensor which results in a similar diffusion behavior.

Table 7.5: Comparison between EED with $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$, $t = 100$ and CED using $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 1$, $\tau = 0.25$, $t = 100$. The table corresponds to Figure 7.5.

Method	Preference	Precision	Recall	F-Measure
Original	–	0.456	0.464	0.460
EED	no	0.000	0.000	0.000
CED	no	0.000	0.000	0.000
EED	yes	0.562	0.573	0.567
CED	yes	0.562	0.573	0.567

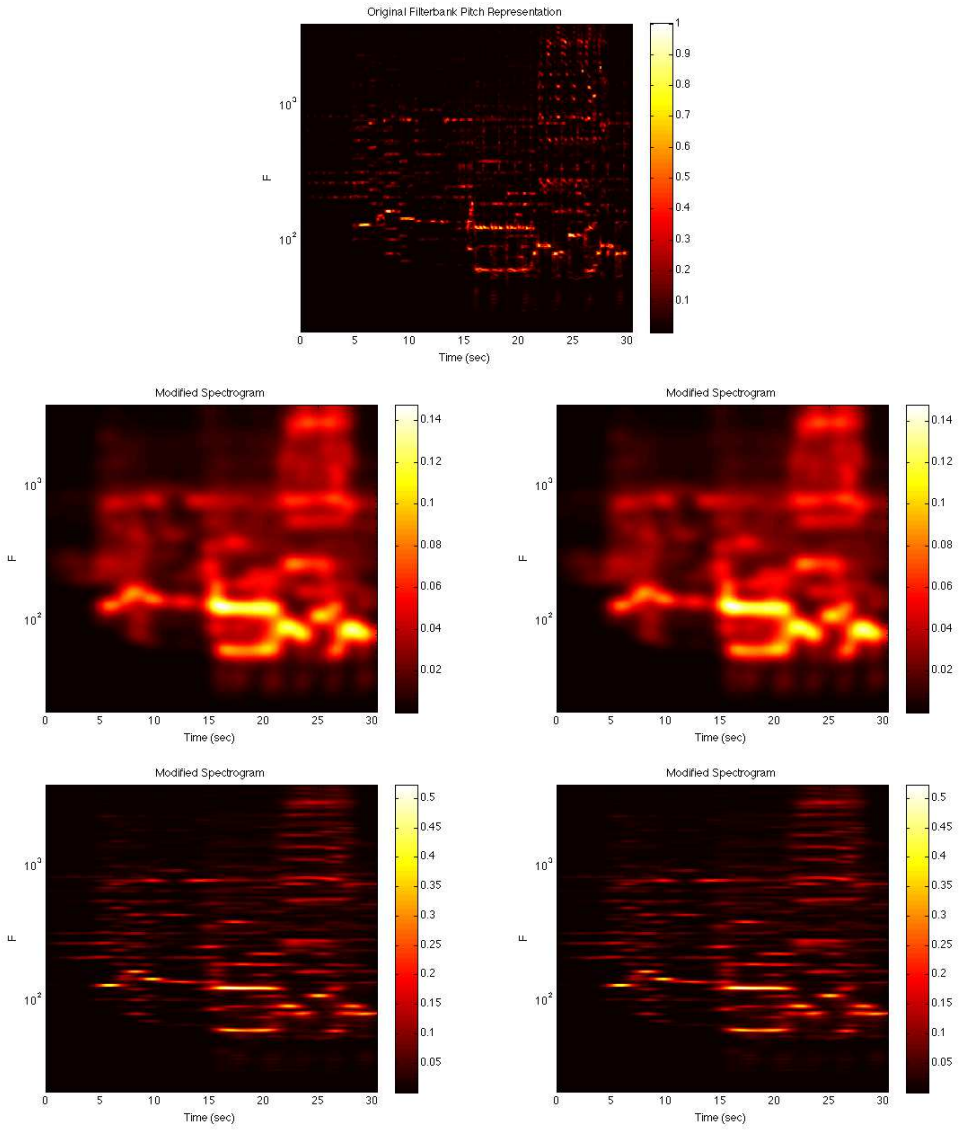

Figure 7.5: Comparison between EED and CED. **Top:** Original spectrogram of "I Am The Walrus" by The Beatles (number 8 in Table A.1). **Middle left:** Edge-enhancing diffusion with $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$, $t = 100$ and no preference for horizontal diffusion. **Bottom left:** Same setting with preference for horizontal diffusion. **Middle right:** Coherence-enhancing diffusion with $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 1$, $\tau = 0.25$, $t = 100$ and no preference for horizontal diffusion. **Bottom right:** Same parameters, but directional preference for horizontal diffusion turned on.

7.2 Morphology

This section deals with morphological operations. After experiments that show how the basic building blocks of morphology, i.e. dilation and erosion as well as opening and closing, work on spectrograms we proceed by focussing on skeletonization where the testfile “Baby’s In Black” by The Beatles (depicted in Figure 7.6) is used.

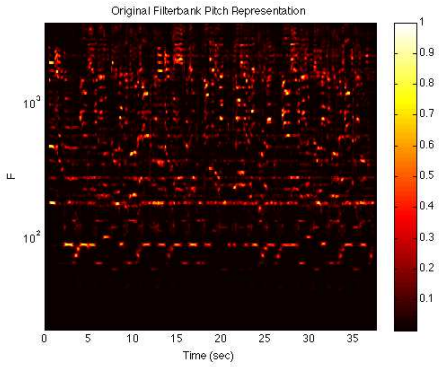

Figure 7.6: Spectrogram of “Baby’s In Black” by The Beatles (number 4 in Table A.1).

7.2.1 Dilation and Erosion

First, we start by investigating the building blocks dilation and erosion using a rectangle and a disk as structuring element. The top row of Figure 7.7 depicts the outcome of dilation and erosion using a rectangle of height 1 and width 10 as structuring element where the top row of Figure 7.8 shows the results for the same computation but with a disk of radius 10 as structuring element. It is clearly visible that dilation blurs the original spectrogram a lot for both structuring elements. While the top left spectrogram in Figure 7.7 allows to guess the original arrangement of frequencies, this information is completely destroyed in the top left spectrogram of Figure 7.8. The erosion which is depicted for a rectangle and a disk as structuring element in the top right of Figure 7.7 and Figure 7.8, respectively, we find out that this operation throws a lot of information away. Similarly to dilation with a disk, erosion with a disk structuring element degenerates the final result in such a way that it is practically not useful. In contrast, erosion with a rectangle removes almost every frequency contribution while preserving the basic structure such that the most important frequencies for chord recognition are still available, compare Table 7.6.

Generally speaking, using a disk as structuring element is not an optimal choice since a disk destroys – by virtue of its shape – the frequency distribution in the spectrogram. This fact is also supported by the quality measure, see Table 7.7.

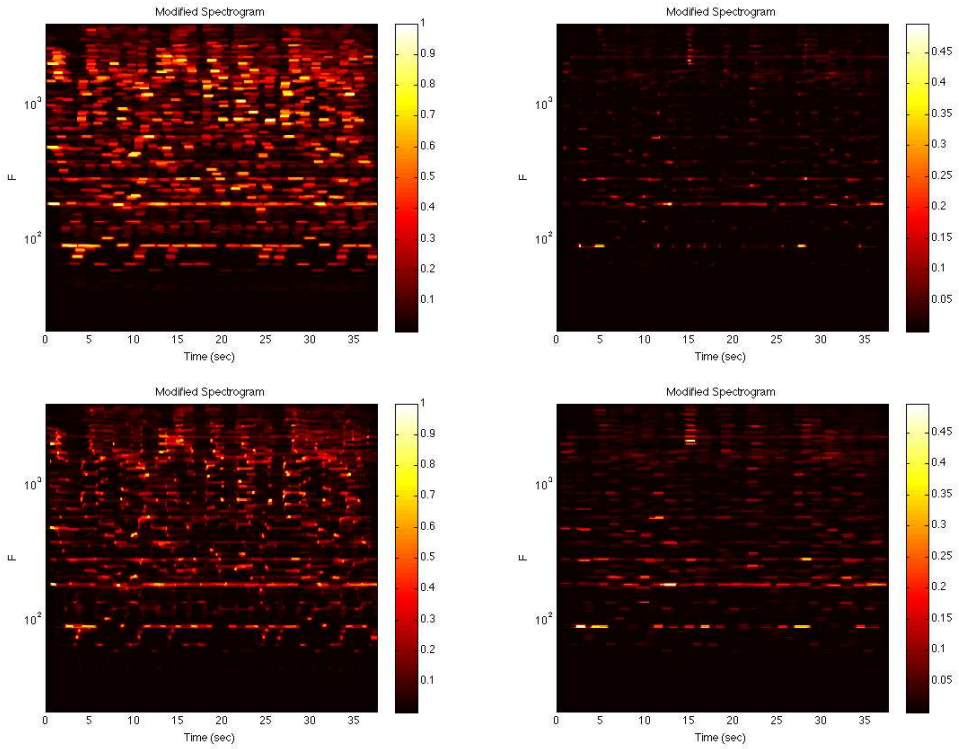

Figure 7.7: Morphological operations on the spectrogram of "Baby's In Black" by The Beatles with a rectangle of height 1 and width 10 as structuring element. **Top left:** Dilation. **Top right:** Erosion. **Bottom left:** Closing. **Bottom right:** Opening.

Table 7.6: Morphological operations on the spectrogram of "Baby's In Black" by The Beatles with a rectangle of height 1 and width 10 as structuring element. See also Figure 7.7.

Method	Precision	Recall	F-measure
Original	0.475	0.482	0.479
Dilation	0.438	0.445	0.442
Erosion	0.714	0.725	0.720
Closing	0.505	0.512	0.509
Opening	0.722	0.733	0.727

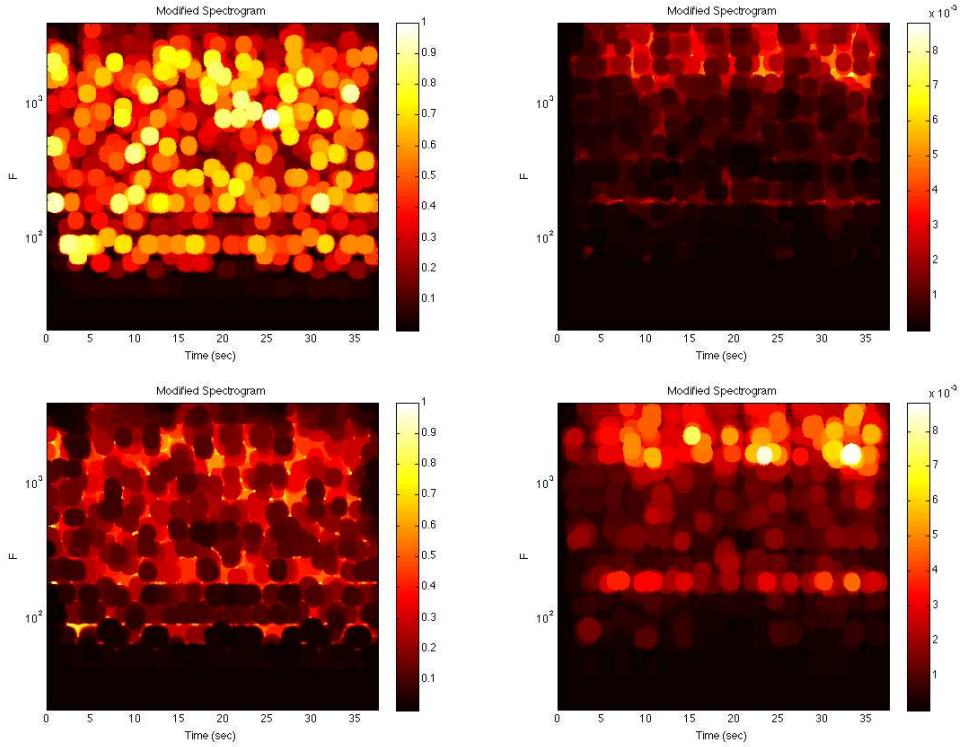

Figure 7.8: Morphological operations on the spectrogram of "Baby's In Black" by The Beatles with a disk of radius 10 as structuring element. **Top left:** Dilation. **Top right:** Erosion. **Bottom left:** Closing. **Bottom right:** Opening.

Table 7.7: Morphological operations on the spectrogram of "Baby's In Black" by The Beatles with a disk of radius 10 as structuring element. See also Figure 7.8.

Method	Precision	Recall	F-measure
Original	0.475	0.482	0.479
Dilation	0.039	0.040	0.040
Erosion	0.017	0.018	0.017
Closing	0.204	0.207	0.206
Opening	0.067	0.068	0.067

7.2.2 Opening and Closing

Figure 7.7 shows on the bottom the spectrograms resulting from applying closing and opening to the spectrogram given in Figure 7.6. Let us recall that a closing operation is a dilation followed by an erosion and an opening operation is the opposite (first erosion, then dilation). Please note that viewing the spectrograms in Figure 7.7 column-wise gives in the top row automatically the intermediate step for the operation depicted in the bottom row.

We investigate the closing in Figure 7.7 first. The blurring introduced by dilation is slightly removed by the followed erosion resulting in an improvement of the outcome, compare Table 7.6. We have seen before that applying erosion performs well for the chord recognition process since it removes mainly noisy frequency contributions. Applying a dilation to the erosion step brings some contributions back. This can be seen in the bottom right of Figure 7.7 where the opening on the spectrogram given in Figure 7.6 is shown. The improvement of the opening over simply performing erosion is also supported by the values precision, recall and F-measure given in Table 7.6.

As done in the section before, we briefly touch the result computed with the disk structuring element as given in Figure 7.8. In this case we see that closing and opening cannot improve the result very much. In fact, the results in Table 7.7 show that opening and closing bring enhancements to the results computed by an erosion or dilation, but still do not improve the final result compared to the original. This circumstance is due to the situation already mentioned before, i.e. that the disk shape destroys too much information about the frequency distribution.

7.2.3 Skeletonization

In the following, we focus on the *parabolic skeletonization* algorithm introduced in Section 6.2 and we refer to it by simply using *skeletonization*. Thus, let us first show the application of parabolic dilation and parabolic erosion on “Baby’s In Black” by The Beatles. The processed spectrograms are given in Figure 7.9. We immediately notice that neither the application of parabolic dilation nor the application of parabolic erosion results in such extreme changes of the structure in the spectrograms as investigated in Figures 7.7 and 7.8. Both, parabolic dilation and parabolic erosion, blur the frequencies marginally where a small blow-up can be noticed for parabolic dilation and a small shrinkage for parabolic erosion while basically preserving the overall structure of the spectrogram.

MATLAB comes with a built-in skeletonization function (`bwmorph`) that only works on binary images. The problem, though, with this approach is to find a suitable threshold to compute a binary version of the grayscale image. MATLAB also provides a function for this task (`graythresh`) which computes a global image threshold following the method proposed by N. Otsu [27]. The binary image can be computed using `im2bw` (also a MATLAB function). Furthermore, the result can be slightly enhanced by using the color information of the original spectrogram at the skeleton points. Figure 7.10 shows the binary image of the “Baby’s In Black” spectrogram as well as its skeleton computed by `bwmorph`. The result of this procedure is a skeleton with thin, sharp edges which is usually disadvantageous. Music

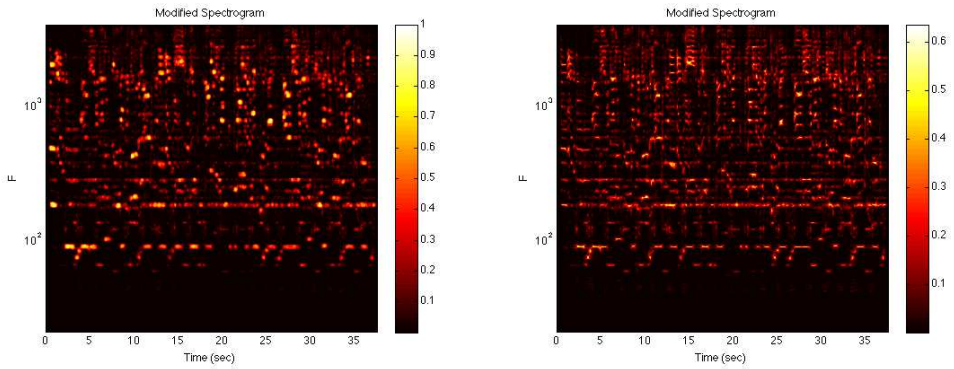

Figure 7.9: Parabolic morphological operations on the spectrogram of "Baby's In Black" by The Beatles. **Left:** Parabolic dilation with $s_x = 0.1$ and $s_y = 0.1$. **Right:** Parabolic erosion with $s_x = 0.1$ and $s_y = 0.1$.

is dependent on things like overtones that are removed by the application of `graythresh`. Due to this fact, the result is – though it might be of good visual quality – not well suited for chord recognition which can also be seen in Table 7.8.

Figure 7.11 compares the skeleton computed with `bwmorph` plus additional color information with the skeletons resulting from the application of the parabolic skeletonization algorithm with different values for s_x and s_y. The difference between the `bwmorph` skeleton and the parabolic ones is quite obvious: the skeletons computed with parabolic skeletonization introduce some blurring while the `bwmorph` skeleton has thin, sharp edges. As already mentioned, the parabolic skeletons preserve information necessary for music information retrieval. Comparing these skeletons among each other, hardly any difference can be seen. The skeleton produced with $s_x = 1$ and $s_y = 0.1$ shows a little preference for horizontal structures where the result using $s_x = 0.1$ and $s_y = 1$ gives more a vertical preference.

Due to the quality measure, the parabolic skeletonization with $s_x = 0.1$ and $s_y = 0.1$ gives the best result for this testfile, compare Table 7.8. However, the outcome depends on the input data. For some files, an improved result for $s_x = 1$ and $s_y = 0.1$ can be seen, but in general, $s_x = s_y = 0.1$ is a reasonable starting point.

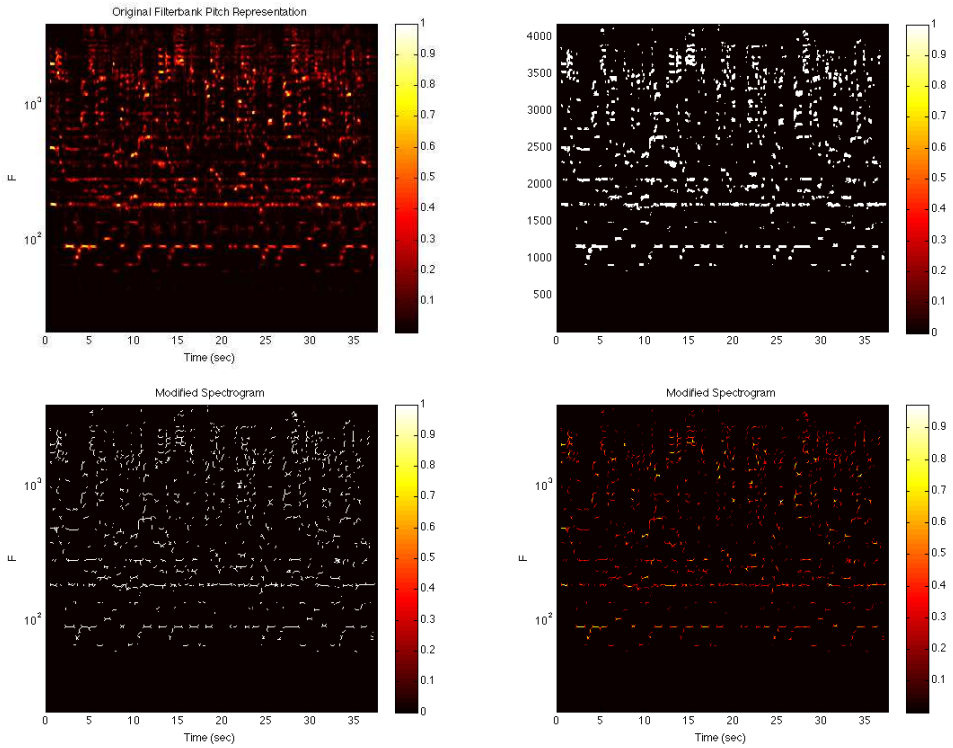

Figure 7.10: Skeletonization of "Baby's In Black" by The Beatles. **Top left:** Original spectrogram. **Top right:** Binary image of the original spectrogram generated using the built-in MATLAB function `im2bw` with an automatically chosen threshold realized by the built-in function `graythresh`. **Bottom left:** Skeleton computed with the MATLAB function `bwmorph` using the binary version of the original spectrogram. **Bottom right:** Same skeleton with color information of the original spectrogram at skeleton points.

Table 7.8: Comparison of skeletonization algorithms on "Baby's In Black" by The Beatles. The first two skeletonizations correspond to the `bwmorph` function that is built into MATLAB. The binary image for `bwmorph` was generated using `im2bw` with an automatically chosen threshold realized by `graythresh`. Last three are parabolic skeletonizations with a threshold of 0.001 for different values for s_x and s_y.

Method	s_x	s_y	Precision	Recall	F-measure
Original	–	–	0.475	0.482	0.479
MATLAB	–	–	0.419	0.425	0.422
MATLAB + color	–	–	0.433	0.440	0.437
Parabolic	0.1	0.1	0.537	0.545	0.541
Parabolic	1	0.1	0.515	0.522	0.519
Parabolic	0.1	1	0.433	0.440	0.437

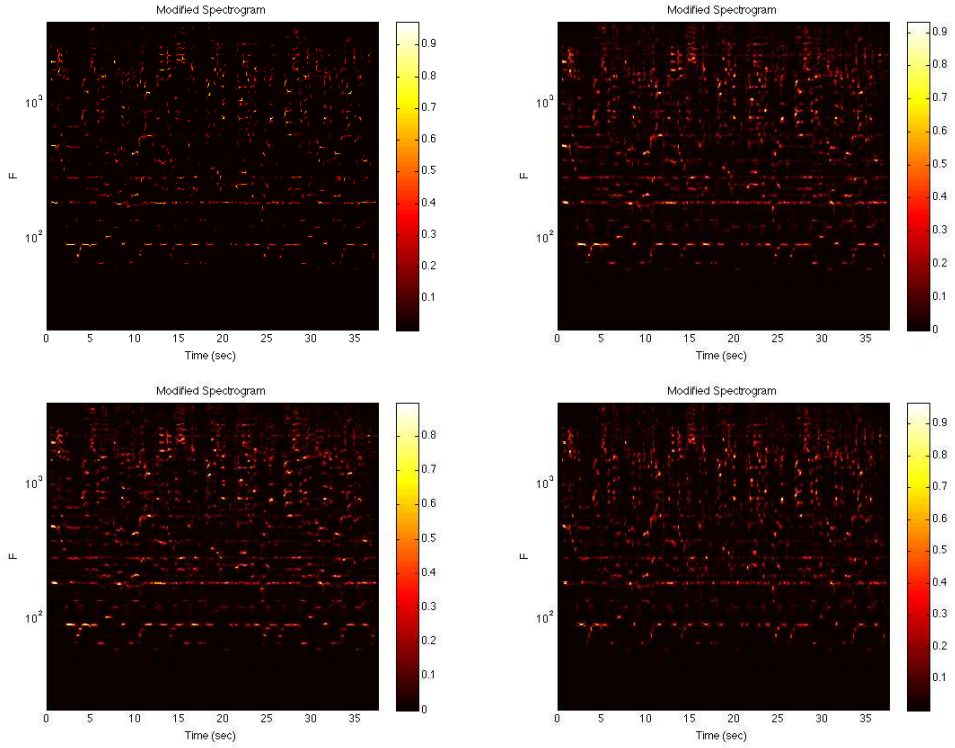

Figure 7.11: Different skeletonization approaches on "Baby's In Black" by The Beatles. **Top left:** Skeleton computed with `bwmorph` and additional color information. **Top right:** Parabolic skeletonization with $s_x = 0.1$, $s_y = 0.1$ and a threshold of 0.001. **Bottom left:** Same, but $s_x = 1$, $s_y = 0.1$. **Bottom right:** Same, but $s_x = 0.1$, $s_y = 1$.

7.3 NL-Means

In Sections 5.2 and 6.3, we have seen that various improvements can be made to NL-means to make the algorithm faster. These improvements also introduce assumptions and modifications to the originally intended algorithm. We will investigate in this section how these assumptions and changes influence the outcome of the NL-means algorithm for chord recognition. In our experiments, we omit the usage of the Gaussian convolution of the Euclidean distance as shown in equation (5.2). The influence of this modification on the result is negligible because the Gaussian convolution only slightly modifies the NL-means filter. In addition, the runtime of the algorithm is a bit lower. Due to the computational complexity, we do not consider the basic version where the search space for similar patches is not restricted.

In the following – and also in Chapter 8 – we stick to the term *NL-means* if we do not consider the modifications presented in Section 5.2.2 and to *fast NL-means* if the modified weighting function as given in equation (5.9) is used. For both approaches, NL-means and fast NL-means, we consider all other improvements such as the locally adaptive weighting and the modified search and similarity windows.

7.3.1 Improved NL-means

We start with the comparison between NL-means with and without adaptive weighting (see Section 5.2.3) for different parameters λ.

No directional preference. In this experiment, no directional preference – i.e. a quadratic similarity window and a quadratic search window – is used. Figure 7.12 shows the spectrograms of "Tell Me Why" by The Beatles. In the left column of the figure, we see the resulting spectrograms for different values of λ without adaptive weighting. In the right column, exactly the same computation was performed but with adaptively changing the values of the distance weight λ according to equation (5.10). It is clearly visible that the algorithm blurs structures more if λ is not chosen adaptively. This coincides with the fact that a globally chosen λ is not always preferable, see also [7, 34]. The quality measure given in Table 7.9 shows that for $\lambda = 10$ this does not hold for the examined testfile, but in general, an adaptively chosen distance weight λ gives either the same result or improves it in comparison to the version without adaptation.

The best result in Figure 7.12 is achieved for $\lambda = 1$ in conjunction with adaptive weighting which can also be visually verified. Fine structures like small contributions in the top quarter of the spectrogram are enhanced pretty well compared to the version without adaptive weighting where detailed structures are unsharp after the application of the algorithm due to the heavy blurring involved. Moreover, points where no or small contributions can be found – i.e. the dark parts in the spectrograms – are filled up with contributions the higher λ gets.

Horizontal directional preference. In addition to the previous experiment, we have a look at how a directional preference – i.e. a rectangular similarity window (compare Section 6.3) –

Table 7.9: Comparison of choosing λ adaptively or not for NL-means with a 21×21 search window and a 7×7 similarity window. This table corresponds to Figure 7.12.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.213	0.221	0.217
1	no	0.142	0.147	0.145
1	yes	0.256	0.266	0.261
5	no	0.062	0.064	0.063
5	yes	0.071	0.074	0.072
10	no	0.059	0.061	0.060
10	yes	0.049	0.051	0.050

changes the result. We stick to the same experimental setting as in the experiment before, but change the height of the similarity window to 3, i.e. we use a similarity window of size 7×3. The spectrograms belonging to this experiment are given in Figure 7.13. Regarding this figure, the first thing we notice is that horizontal structures dominate. Let us compare Tables 7.9 and 7.10. First, improvements are achieved when using a rectangular window in conjunction with the usage of adaptive weighting. Second, for the setting where adaptive weighting is not used, no differences between a 7×7 and a 7×3 similarity window can be found. An exception for both cases is $\lambda = 1$.

The main reason to only restrict the similarity window – i.e. the inpainting window – is to make the search for similar structures in the spectrogram more reliable. Let us assume for the moment a 21×3 search window and a 7×3 similarity window. We shortly give here the corresponding values for precision, recall and F-measure for $\lambda = 1$. For the variant where λ is not computed adaptively we have $(P, R, F) = (0.130, 0.135, 0.132)$ and for the variant with adaptive weighting, we have $(P, R, F) = (0.198, 0.205, 0.201)$. Comparing these values with the corresponding ones in Table 7.10, we notice higher values for precision, recall and F-measure, meaning that the result computed with a 21×21 search window is advantageous.

Table 7.10: Comparison of choosing λ adaptively or not for NL-means with a 21×21 search window and a 7×3 similarity window. This table corresponds to Figure 7.13.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.213	0.221	0.217
1	no	0.151	0.157	0.154
1	yes	0.216	0.224	0.220
5	no	0.062	0.064	0.063
5	yes	0.114	0.119	0.116
10	no	0.059	0.061	0.060
10	yes	0.068	0.071	0.069

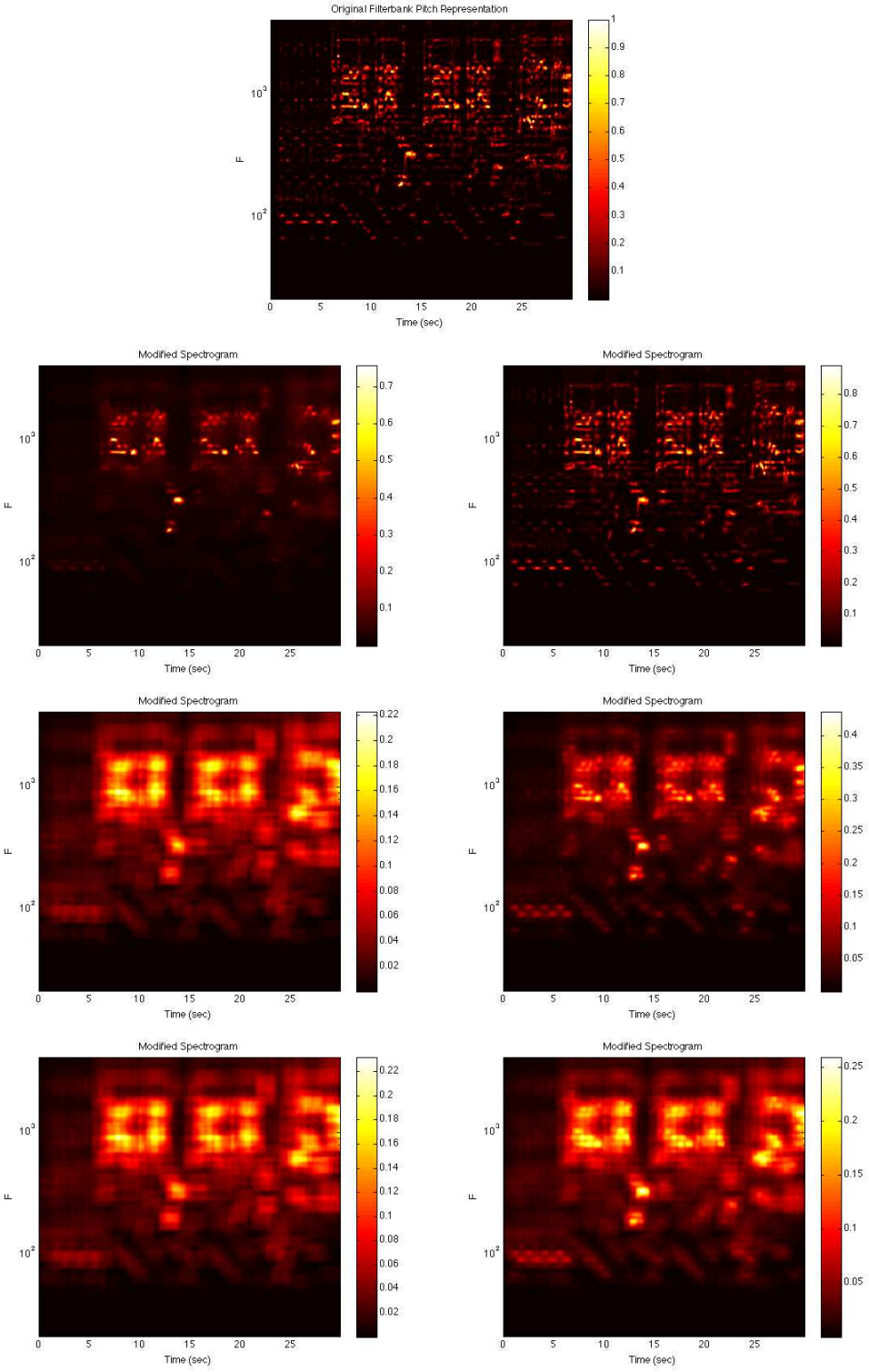

Figure 7.12: Comparison of choosing λ adaptively or not for NL-means using the test-file "Tell Me Why" by The Beatles (number 12 in Table A.1). All spectrograms were computed using a 21×21 search window and a 7×7 similarity window. **Top:** Original spectrogram. **Left:** λ is not chosen adaptively. **Right:** λ is chosen adaptively. **Middle to bottom:** $\lambda = 1, 5, 10$.

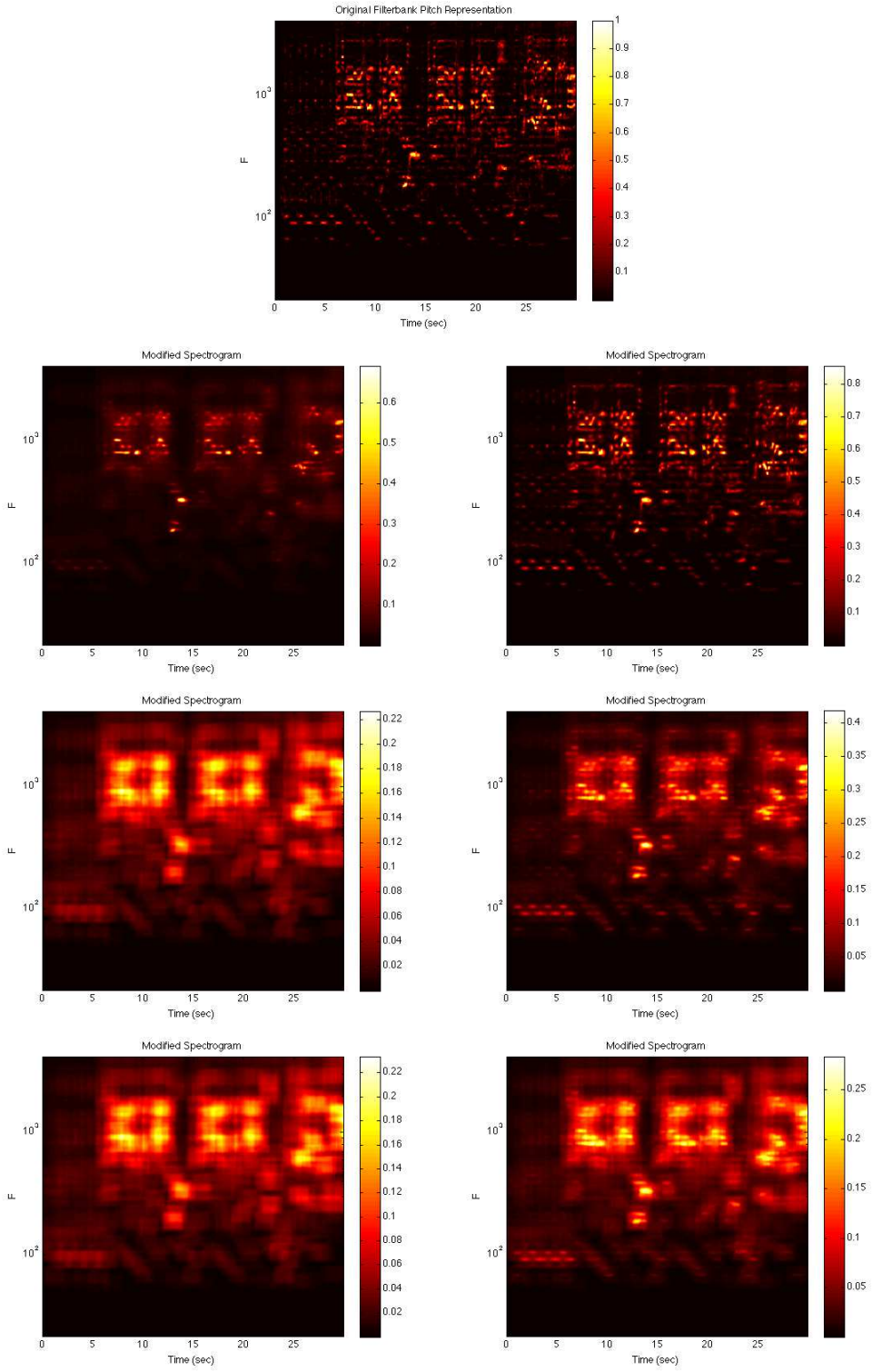

Figure 7.13: Same as in Figure 7.12, but with a 21×21 search window and a 7×3 similarity window.

7.3.2 Fast NL-means

In this paragraph, let us discuss the fast NL-means algorithm (compare Section 5.2.2) with and without adaptive weighting for different λ.

No directional preference. We start again with a quadratic window for the search as well as for the similarity window. Furthermore, we investigate how the two parameters η_1 and η_2 change the result. These parameters are a crucial threshold in the fast NL-means algorithm which can be seen in equation (5.9).

In Figure 7.14, the processing of "Tell Me Why" (number 12 in Table A.1) with fast NL-means is shown. To compute these spectrograms, $\eta_1 = 0.9$ and $\eta_2 = 1.1$ are chosen. Except for the computation with $\lambda = 1$ and adaptive weighting – which enhances the result compared to the original – all the other parameter settings did not improve the spectrogram at all, see Table 7.11. It is also clearly visible that in all the spectrograms that did not enhance the original a lot of blurring is present that does not preserve the fine detailed structures.

Table 7.11: Comparison of choosing λ adaptively or not for fast NL-means with a 21×21 search window, a 7×7 similarity window, $\eta_1 = 0.9$ and $\eta_2 = 1.1$. This table corresponds to Figure 7.14.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.213	0.221	0.217
1	no	0.179	0.186	0.182
1	yes	0.253	0.263	0.258
5	no	0.142	0.147	0.145
5	yes	0.160	0.167	0.164
10	no	0.142	0.147	0.145
10	yes	0.142	0.147	0.145

Figure 7.15 shows the same experiments by using a 21×21 search window and a 7×7 similarity window where $\eta_1 = 0.5$ and $\eta_2 = 1.5$ are used. Intuitively, we know that similar neighborhoods should have similar gray values. Increasing the range given in equation (5.4) means that we are more tolerant when it comes to deviations from the average gray value in the two windows that are compared. Let us investigate the spectrograms in Figure 7.14 and the corresponding spectrograms in Figure 7.15. We observe that for $\eta_1 = 0.5$ and $\eta_2 = 1.5$ the spectrograms are much more dense (smeared) in comparison to the ones computed with $\eta_1 = 0.9$ and $\eta_2 = 1.1$. This visual observation is also supported by the measurement of precision, recall and F-measure given in Table 7.11 and 7.12 where the result has in general a higher F-measure for $\eta_1 = 0.9$ and $\eta_2 = 1.1$ except for $\lambda = 1$ with adaptive weighting where the version with $\eta_1 = 0.5$ and $\eta_2 = 1.5$ gives a slight improvement. Note that choosing η_1 and η_2 further away from 1 – as in the setting with $\eta_1 = 0.5$ and $\eta_2 = 1.5$ – may increase the quality of the denoised version but can also introduce a higher error since more contributions are considered to compute the averaged weighting.

Table 7.12: Comparison of choosing λ adaptively or not for fast NL-means. A 21×21 search window, a 7×7 similarity window and $\eta_1 = 0.5$, $\eta_2 = 1.5$ is used. This table corresponds to Figure 7.15.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.213	0.221	0.217
1	no	0.173	0.179	0.176
1	yes	0.256	0.266	0.261
5	no	0.117	0.122	0.119
5	yes	0.136	0.141	0.138
10	no	0.108	0.112	0.110
10	yes	0.120	0.125	0.123

Horizontal directional preference. We finally look at the same setting as in the experiments described before but use a rectangular 7×3 similarity window, see Figures 7.16 and 7.17 with the corresponding Tables 7.13 and 7.14.

For a rectangular window, we observe exactly the opposite in comparison to a quadratic one. For the combination of $\eta_1 = 0.5$ and $\eta_2 = 1.5$, the results are advantageous in comparison to the ones computed with $\eta_1 = 0.9$ and $\eta_2 = 1.1$. The reason for this issue can be explained with the following intuition.

So far, we have seen several ingredients that influence the outcome of the NL-means algorithm: the choice of λ which controls the weighting, the change of the similarity window to only perform inpainting with a preference for a certain direction and the *fast* NL-means algorithm that only uses the weighting function under certain circumstances which is dependent – among other conditions – on the choice of η_1 and η_2. Making the interval given in (5.4) bigger also means to include more pixels in the computation of the weighted averaging since the algorithm is more tolerant concerning deviations from the average grayvalue of the two windows that are compared. In some cases, narrowing the interval in (5.4) and using a rectangular similarity window at the same time results in too few information for the inpainting step. On the other hand, including too much information might also not be useful for the denoising process.

This behavior described above cannot be seen with every testfile. It can also be observed that for some input data the outcome is enhanced more for $\eta_1 = 0.9$ and $\eta_2 = 1.1$. Since the result in this example does not differ much for the settings $\eta_1 = 0.9, \eta_2 = 1.1$ and $\eta_1 = 0.5, \eta_2 = 1.5$, we stick to $\eta_1 = 0.9$ and $\eta_2 = 1.1$.

Table 7.13: Comparison of choosing λ adaptively or not for fast NL-means with $\eta_1 = 0.9$ and $\eta_2 = 1.1$ where a 21×21 search window and a 7×3 similarity window is used. This table corresponds to Figure 7.16.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.213	0.221	0.217
1	no	0.253	0.263	0.258
1	yes	0.235	0.244	0.239
5	no	0.231	0.240	0.236
5	yes	0.238	0.247	0.242
10	no	0.228	0.237	0.233
10	yes	0.235	0.244	0.239

Table 7.14: Comparison of choosing λ adaptively or not for fast NL-means with $\eta_1 = 0.5$ and $\eta_2 = 1.5$. Furthermore, a 21×21 search window and a 7×3 similarity window is used. This table corresponds to Figure 7.17.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.213	0.221	0.217
1	no	0.238	0.247	0.242
1	yes	0.225	0.234	0.230
5	no	0.247	0.256	0.252
5	yes	0.241	0.250	0.245
10	no	0.247	0.256	0.252
10	yes	0.247	0.256	0.252

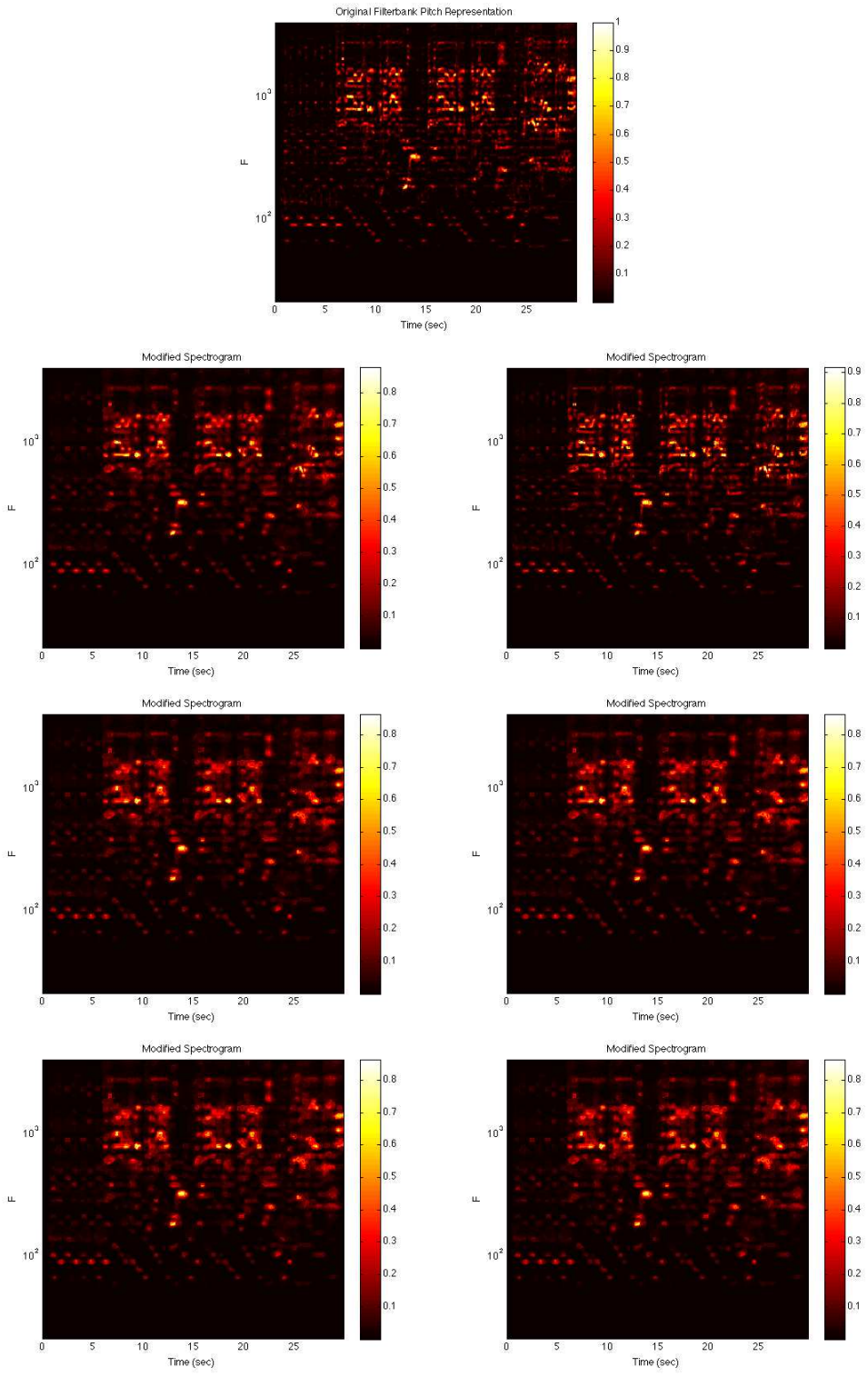

Figure 7.14: Comparison of choosing λ adaptively or not for fast NL-means using the testfile "Tell Me Why" by The Beatles (number 12 in Table A.1). All spectrograms were computed using a 21×21 search window, a 7×7 similarity window, $\eta_1 = 0.9$ and $\eta_2 = 1.1$. **Top:** Original spectrogram. **Left:** λ is not chosen adaptively. **Right:** λ is chosen adaptively. **Middle to bottom:** $\lambda = 1, 5, 10$.

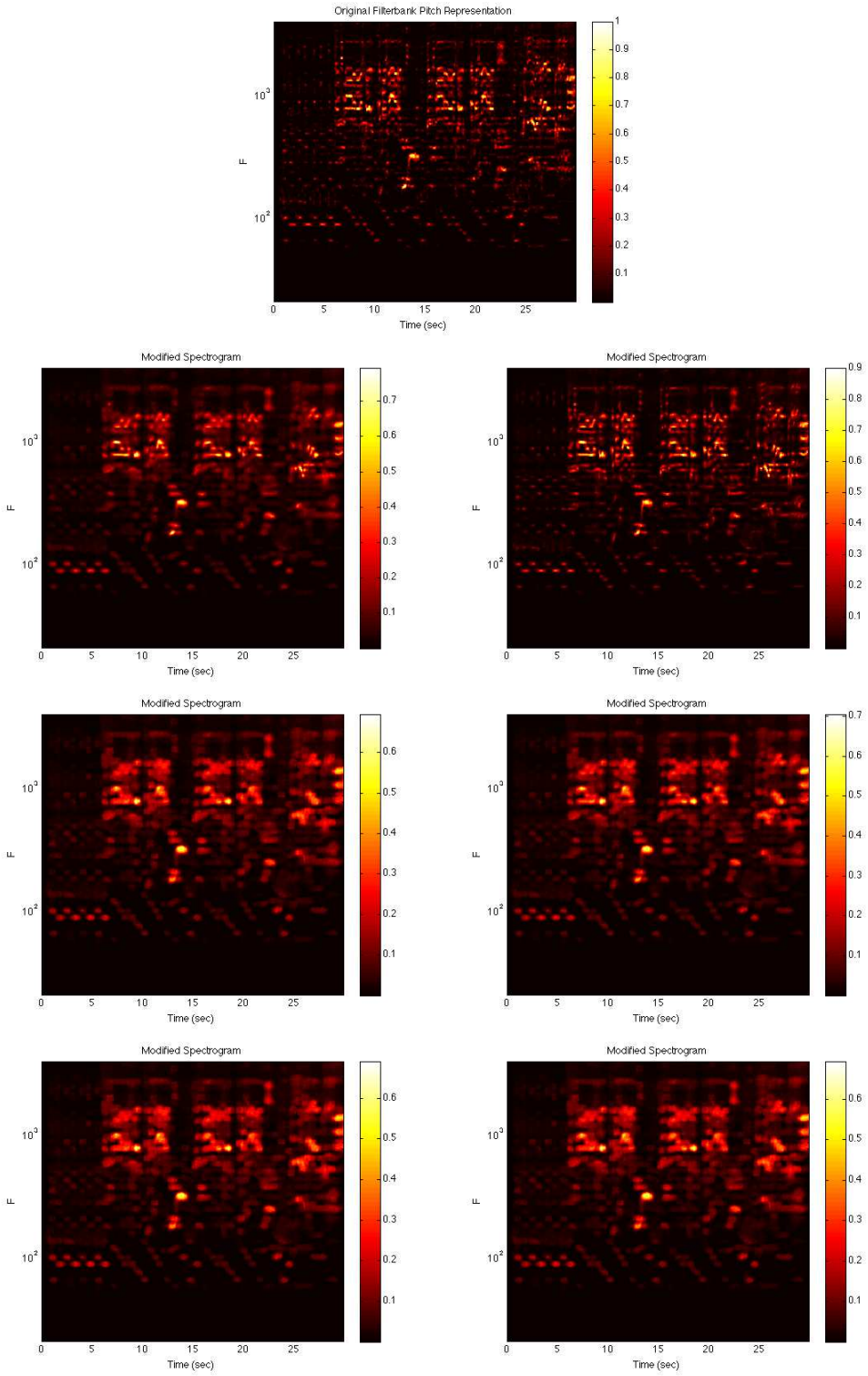

Figure 7.15: Same as in Figure 7.14, but with $\eta_1 = 0.5$ and $\eta_2 = 1.5$.

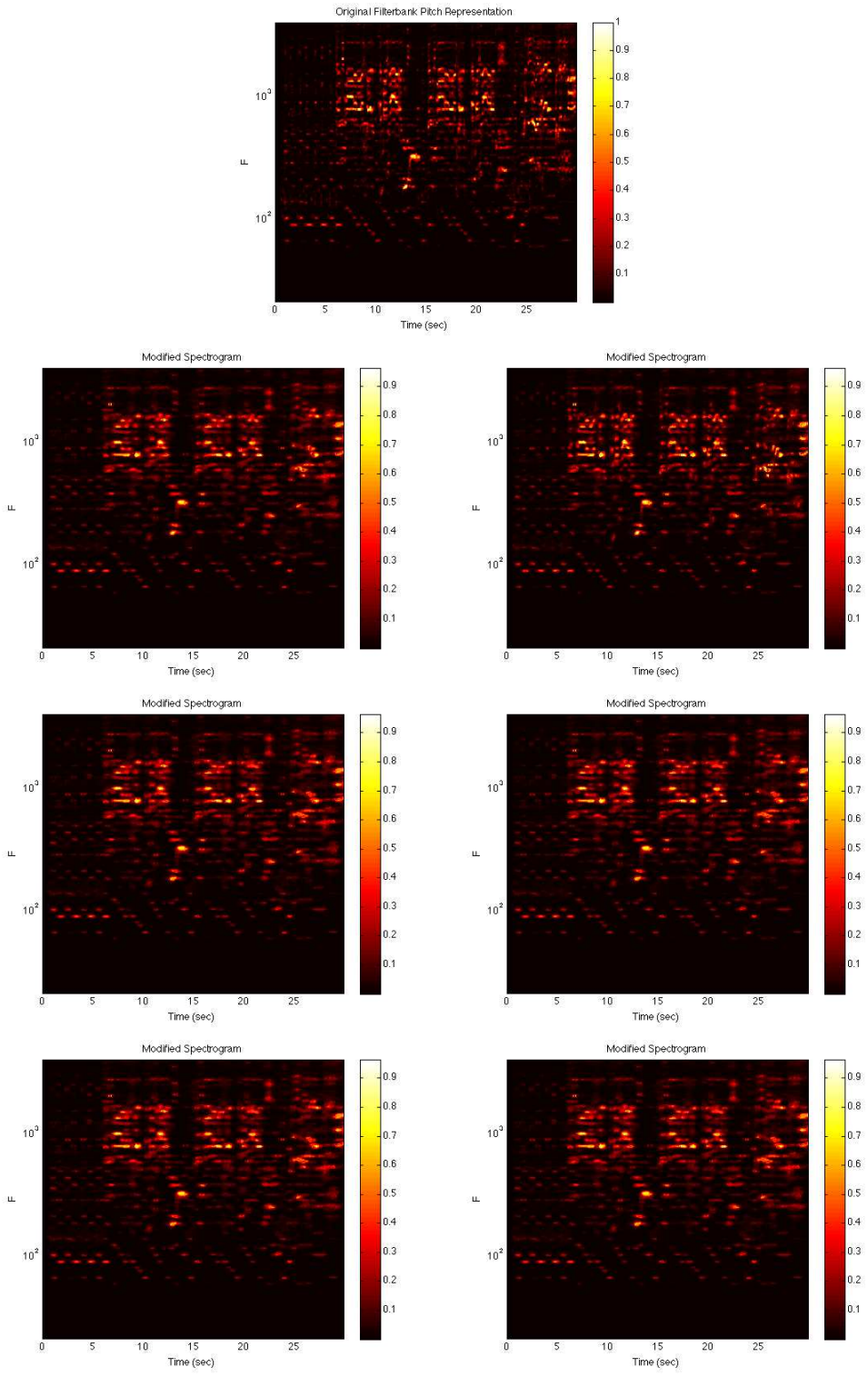

Figure 7.16: Comparison of choosing λ adaptively or not for fast NL-means using the testfile "Tell Me Why" by The Beatles (number 12 in Table A.1). All spectrograms were computed using a 21×21 search window, a 7×3 similarity window, $\eta_1 = 0.9$ and $\eta_2 = 1.1$. **Top:** Original spectrogram. **Left:** λ is not chosen adaptively. **Right:** λ is chosen adaptively. **Middle to bottom:** $\lambda = 1, 5, 10$.

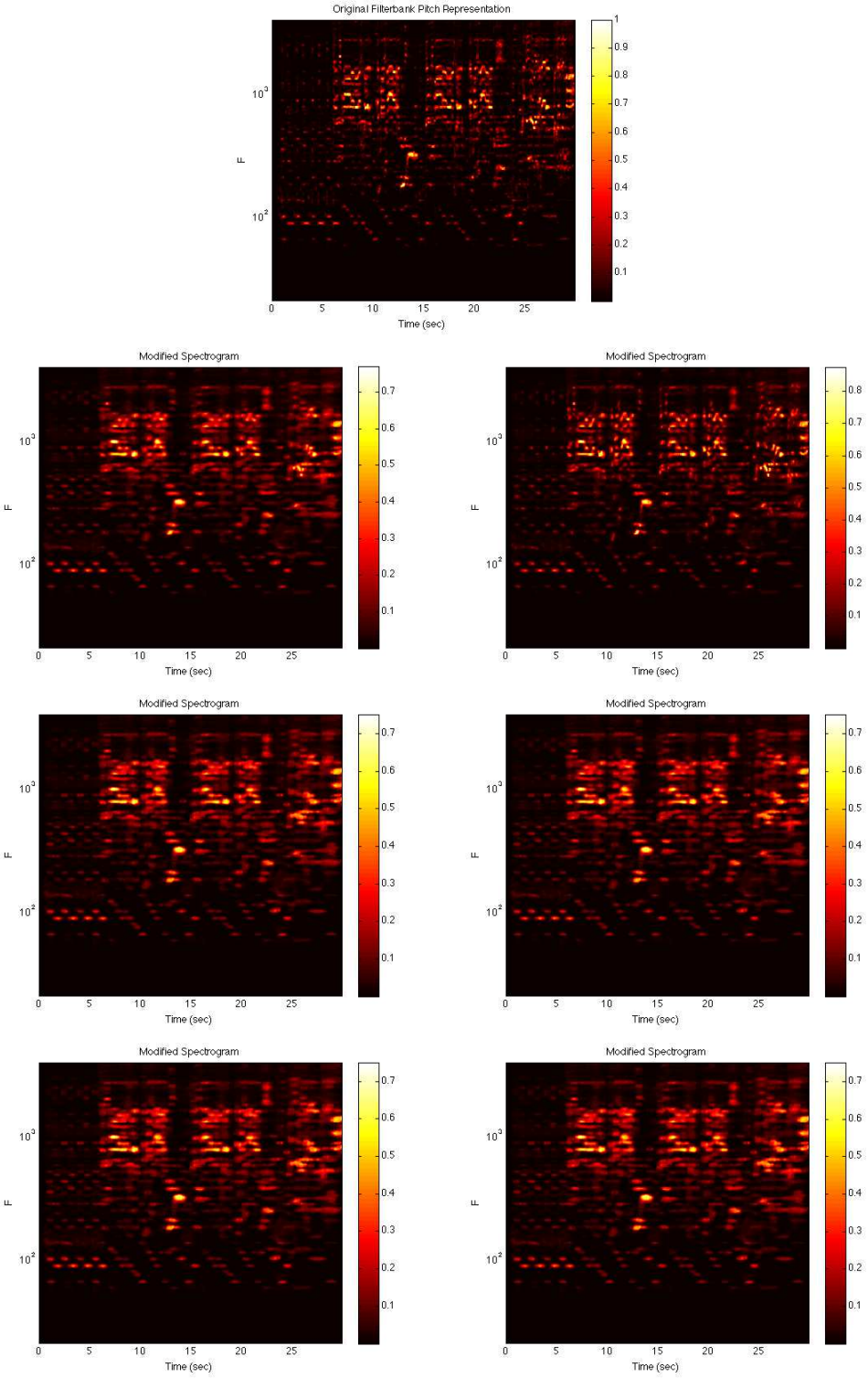

Figure 7.17: Same as in Figure 7.16, but with $\eta_1 = 0.5$ and $\eta_2 = 1.5$.

7.4 Discussion and Summary

This section concludes the experiments on chord recognition by testing a standard parameter set for EED, CED, skeletonization, NL-means and fast NL-means on all the testfiles. This is useful since in real-world examples, a ground truth is generally not available. Thus, having a reasonable starting point is crucial. Furthermore, standardized parameter settings are important for the automated application of the methods. Since the impact of the different methods is visually demonstrated in the previous sections, only the tables giving precision, recall and F-measure are proposed here.

For the computations, the following parameter sets are applied and preference for horizontal structures is always used except for skeletonization.

- EED:
  $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$, $t = 100$.
- CED:
  $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 0.01$, $\tau = 0.25$, $t = 1000$.
- Skeletonization:
  $s_x = 0.1$, $s_y = 0.1$, threshold = 0.001.
- NL-means:
  21×21 search window, 7×3 similarity window, $\lambda = 1$, with adaptive weighting.
- Fast NL-means:
  21×21 search window, 7×3 similarity window, $\eta_1 = 0.9$, $\eta_2 = 1.1$, $\lambda = 1$, with adaptive weighting.

Performance comparison across all approaches. In the following, we state the results of the different methods applied to all testfiles with the use of precision, recall and F-measure. The quality of the algorithmic treatment is given for each file in Tables 7.15 to 7.27.

As a general trend, we note that diffusion processes outperform skeletonization and NL-means. The testfiles “Helter Skelter” and “Tell Me Why” are the only exceptions to this finding, compare Tables 7.21 and 7.26, where none of the diffusion processes could beat one of the other algorithms. On “Cry Baby Cry” and “You Can’t Do That” (see Tables 7.20 and 7.27) at least one of the diffusion processes was superior to all other algorithms. Furthermore, we find that skeletonization beats both NL-means approaches on several testfiles (numbers 1, 3, 4, 6, 7 and 13). In addition, there exist cases where skeletonization is superior to NL-means, but not fast NL-means, see Tables 7.16, 7.19, 7.24 and 7.25, and on one of the testfiles, the application of skeletonization gives a higher F-measure compared to fast NL-means, but not NL-means (Table 7.22).

Within the class of NL-means, we notice that the fast NL-means algorithm is superior to the NL-means algorithm in almost every case. This observation supports the theory that too much information leads to a significant error in the weighted averaging. The only exception

is the file “I Am The Walrus”, compare Table 7.22. In general, the fast NL-means algorithm uses the weighting function in comparison to NL-means less often. For this concrete example this may result in too few data for the inpainting step since the spectrogram is sparse in the first half (0 to about 15 seconds). Thus, every time the weighting function is used, it consists of few contributions from the inpainting neighborhood, i.e. the similarity window.

Runtime considerations. As already stated earlier, the runtime of the algorithms is not the main concern of this thesis. Nevertheless, let us give for the sake of completeness one example. Table 7.28 shows the runtime of the different algorithms on the file “Back In The USSR” (number 5 in Table A.1). Let us mention that NL-means still needs a high runtime even though the search space is restricted by the use of a quadratic search window.

Combination of different filters. It is also natural to combine different methods. Exemplarily, we investigate skeletonization as a preprocessing as well as a postprocessing step to diffusion on the testfile “Cry Baby Cry”. Since edge-enhancing diffusion and coherence-enhancing diffusion lead to approximately the same result under similar parameter settings, we use edge-enhancing diffusion here. For the diffusion process as well as for the skeletonization we stick to the parameters proposed earlier.

The spectrograms of this experiment where skeletonization is applied before and after edge-enhancing diffusion are given in Figure 7.18 and the corresponding values for precision, recall and F-measure in Table 7.29.

Let us have a look at the spectrograms on the bottom of Figure 7.18. The visual quality of both is different. The one on the right is very blurry which is not the case for the spectrogram on the left. Only judging the visual quality, we probably would say that the spectrogram on the left is advantageous compared to the right one. Recalling our observation from Section 7.2.3, we can explain why this is not the case (see also Table 7.29). Skeletonization makes the edges – i.e. frequency contributions – thinner. This also removes some important information about the frequency composition that cannot be repaired by applying a diffusion process afterwards. In contrast, applying the diffusion before skeletonization is desired since the diffusion process enhances structures that should not be enhanced and skeletonization works as a sort of correction step for this issue.

Limitations of chord recognition. To conclude the experiments on chord recognition, we want to mention an interesting point. In every information retrieval task it can happen that the feature extraction might extract something wrong. In the following example, the feature extraction is – informally speaking – both right and wrong: right due to the input data (the spectrogram) and wrong due to acoustical judgement. To illustrate this, we consider “And I Love Her” by The Beatles (number 3 in Table A.1). Here it is convenient to look additionally at the chroma-time representation and chord annotation. Figure 7.19 gives both the original data as well as the processed data. Let us look at the bottom of the figure. This plot shows in a color coding which chords are present at which time. Gray stands for the chords that are annotated, red for the retrieved chords. If both match, the color is white. The numbers on

the y-axis identify the chords where numbers from 1 to 12 stand for C Major to B Major and numbers from 13 to 24 for C Minor to B Minor. Let us look at the time frame 0 to 5 seconds on the bottom right representation in Figure 7.19. The feature extraction detected F♯ Major (chord number 7) where the annotation tells that it should be F♯ Minor (chord number 19). This is one of the worst classification errors that can happen. Please note that this misclassification is not introduced, but slightly enhanced by the diffusion process, see bottom row of Figure 7.19. The question is why chord recognition severely fails here.

To answer this, we have to introduce a little bit of music theory. The chord F♯ Major consists of the notes F♯, A♯ and C♯ where F♯ Minor consists of the notes F♯, A and C♯. The only difference is the note in the middle. This must mean that the feature extraction detected an A♯ instead of A. And in fact, if we have a look at the chroma-time representation of the processed data (middle right in Figure 7.19) at the point in time where the note F♯ is almost represented in white, we notice that the note A♯ has a brighter color than A meaning that it is more intense at this point in time. After listening to the testfile, the answer why chord recognition fails on this file is obvious: at exactly this point, the percussive elements start. The percussion contributes to the frequencies corresponding to A♯ which makes it more intense than A. Thus, the feature extraction doesn't recognize the note A as important as the note A♯ meaning that we do not end up with the notes F♯, A, C♯ as the most important notes at this time but with F♯, A♯, C♯. Thus, the chord recognition algorithm concludes that the chord F♯ Major instead of F♯ Minor is played.

Final remark. As we have seen, image processing methods are able to significantly enhance the spectrograms that were considered in this chapter and thus can contribute fruitfully to the success of chord recognition.

Table 7.15: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “All My Loving” by The Beatles (file number 1 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.413	0.426	0.419
EED	0.557	0.574	0.566
CED	0.469	0.483	0.476
Skeletonization	0.413	0.426	0.419
NL-means	0.402	0.414	0.408
Fast NL-means	0.402	0.414	0.408

Table 7.16: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “All You Need Is Love” by The Beatles (file number 2 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.412	0.416	0.414
EED	0.902	0.910	0.906
CED	0.569	0.575	0.572
Skeletonization	0.446	0.450	0.448
NL-means	0.308	0.311	0.309
Fast NL-means	0.468	0.472	0.470

Table 7.17: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “And I Love Her” by The Beatles (file number 3 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.461	0.484	0.472
EED	0.594	0.623	0.609
CED	0.480	0.503	0.491
Skeletonization	0.458	0.481	0.469
NL-means	0.421	0.442	0.431
Fast NL-means	0.424	0.445	0.434

Table 7.18: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “Baby’s In Black” by The Beatles (file number 4 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.475	0.482	0.479
EED	0.695	0.705	0.700
CED	0.562	0.570	0.566
Skeletonization	0.537	0.545	0.541
NL-means	0.389	0.395	0.392
Fast NL-means	0.456	0.463	0.459

Table 7.19: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “Back In The USSR” by The Beatles (file number 5 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.491	0.495	0.493
EED	0.613	0.619	0.616
CED	0.564	0.570	0.567
Skeletonization	0.500	0.505	0.502
NL-means	0.402	0.406	0.404
Fast NL-means	0.525	0.529	0.527

Table 7.20: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “Cry Baby Cry” by The Beatles (file number 6 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.335	0.335	0.335
EED	0.480	0.480	0.480
CED	0.369	0.369	0.369
Skeletonization	0.394	0.394	0.394
NL-means	0.305	0.305	0.305
Fast NL-means	0.351	0.351	0.351

Table 7.21: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “Helter Skelter” by The Beatles (file number 7 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.141	0.142	0.142
EED	0.144	0.145	0.145
CED	0.154	0.155	0.154
Skeletonization	0.232	0.233	0.233
NL-means	0.110	0.110	0.110
Fast NL-means	0.179	0.180	0.179

Table 7.22: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “I Am The Walrus” by The Beatles (file number 8 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.456	0.464	0.460
EED	0.562	0.573	0.567
CED	0.520	0.529	0.525
Skeletonization	0.483	0.492	0.488
NL-means	0.502	0.511	0.506
Fast NL-means	0.480	0.489	0.485

Table 7.23: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “I’m Only Sleeping” by The Beatles (file number 9 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.352	0.357	0.354
EED	0.344	0.349	0.346
CED	0.386	0.391	0.389
Skeletonization	0.315	0.319	0.317
NL-means	0.323	0.327	0.325
Fast NL-means	0.333	0.338	0.336

Table 7.24: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “Let It Be” by The Beatles (file number 10 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.496	0.496	0.496
EED	0.585	0.585	0.585
CED	0.606	0.606	0.606
Skeletonization	0.504	0.504	0.504
NL-means	0.430	0.430	0.430
Fast NL-means	0.532	0.532	0.532

Table 7.25: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “Savoy Truffle” by The Beatles (file number 11 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.185	0.195	0.190
EED	0.370	0.390	0.380
CED	0.247	0.260	0.253
Skeletonization	0.222	0.234	0.228
NL-means	0.198	0.208	0.203
Fast NL-means	0.244	0.256	0.250

Table 7.26: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for "Tell Me Why" by The Beatles (file number 12 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.213	0.221	0.217
EED	0.142	0.147	0.145
CED	0.204	0.212	0.208
Skeletonization	0.207	0.215	0.211
NL-means	0.216	0.224	0.220
Fast NL-means	0.235	0.244	0.239

Table 7.27: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for "You Can't Do That" by The Beatles (file number 13 in Table A.1).

Method	Precision	Recall	F-measure
Original	0.207	0.211	0.209
EED	0.247	0.252	0.249
CED	0.262	0.267	0.265
Skeletonization	0.253	0.258	0.255
NL-means	0.148	0.151	0.150
Fast NL-means	0.210	0.214	0.212

Table 7.28: Comparison of algorithm runtime for "Back In The USSR" by The Beatles.

Method	Runtime [s]
EED	1.734504
CED	49.354818
Skeletonization	75.949182
NL-means	4757.615534
Fast NL-means	270.311324

Table 7.29: Skeletonization as preprocessing or postprocessing step for diffusion. This table gives the corresponding values for precision, recall and F-measure for the spectrograms given in Figure 7.18.

Application chain	Precision	Recall	F-measure
Original	0.335	0.335	0.335
EED $\rightarrow$ Skeletonization	0.486	0.486	0.486
Skeletonization $\rightarrow$ EED	0.572	0.572	0.572

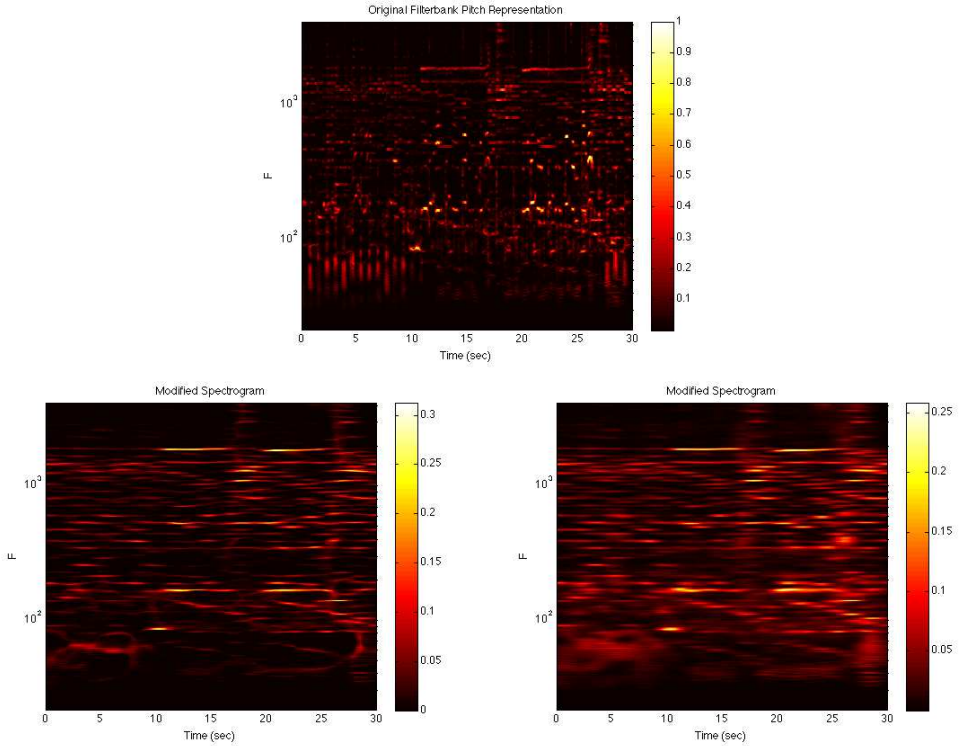

Figure 7.18: Skeletonization as preprocessing or postprocessing step for diffusion. **Top:** Original spectrogram of "Cry Baby Cry" by The Beatles (number 6 in Table A.1). **Bottom left:** Application of a diffusion process followed by a skeletonization. **Bottom right:** Application of a skeletonization followed by a diffusion process.

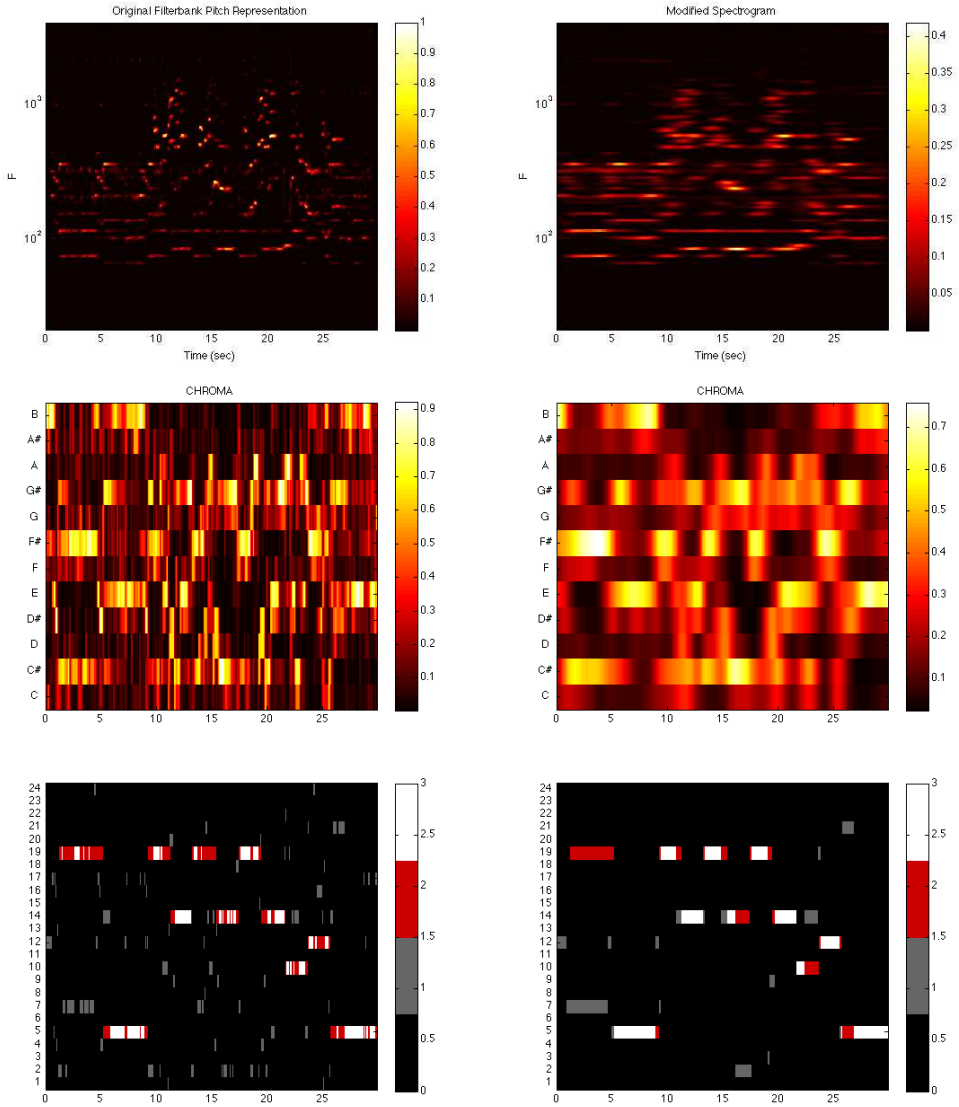

Figure 7.19: Example where chord recognition is fooled by percussion. **Top left:** Original spectrogram of "And I Love Her" by The Beatles (number 3 in Table A.1). **Top right:** Edge-enhancing diffusion with $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$, $t = 80$ and directional preference for horizontal structures. **Middle left:** Chroma-time representation of the original spectrogram. **Middle right:** Corresponding chroma-time representation of the modified spectrogram. **Bottom left:** Representation of the original data that shows the chords that are annotated and the chords that are retrieved in one image. Numbers from 1 to 12 represent the chords from C Major to B Major and from 13 to 24 the chords C Minor to B Minor. Gray color denotes annotated chords, red color retrieved chords. If both match, the color turns white. **Bottom right:** Same representation of the processed data.

Chapter 8

Experiments in Onset Detection

Being able to quit things that don't work is integral to being a winner.
— Tim Ferriss

This chapter describes experiments for the task of onset detection where we are interested in vertical structures. The same experimental setup as described in Chapter 7 is used. Again, the quality of the experimental results is judged by using the concepts introduced in Section 2.3.

As already mentioned in Section 2.2, onset detection is not an easy task. While it seems well-defined at first glance, it is difficult to locate the starting point of a musical note.

For the following experiments, we use the testfiles listed in Table A.2. The testfiles 1 to 8 are excerpts manually annotated by Peter Grosche and Meinard Müller (see [16]). The testfiles 9 to 23 are taken from [4], except for 10, 21 and 22 that are not publicly available, see also [22].

8.1 Diffusion

This section deals with diffusion processes that are performed on the spectrograms for the task of onset detection. Since we are interested in vertical structures, we first compare the results of diffusion processes with and without a directional preference for a fixed number of iteration steps. In a second experiment, we investigate the behavior of the algorithm for different numbers of iterations where d_β is set to 1 and d_α is set to 0 which means that we prefer vertical structures.

8.1.1 Edge-Enhancing Diffusion

Vertical vs. no directional preference. We start by investigating edge-enhancing diffusion on the onset detection problem for the same number of iterations but once with and once without preference for vertical directions. Figure 8.1 depicts the spectrograms of the "Happy Birthday" testfile performed by a choir accompanied by a piano. The spectrogram after applying EED without directional preference is shown on the bottom left where the result of

applying EED with vertical preference is given on the bottom right. The spectrogram on the bottom left shows that diffusion happened also in horizontal direction resulting in the fact that onsets cannot be clearly spotted. Comparing the values for the F-measure in Table 8.1, we observe that the diffusion process without vertical preference slightly enhances the result where the process that diffuses only in vertical direction improves the quality of the computation even further. The investigation of the spectrogram on the bottom right in Figure 8.1 supports this since the structures are sharp and only enhanced in vertical direction. The results given in Table 8.1 coincide with the intuition that a diffusion performed equally in every direction, i.e. a blurring also in the time domain, does not give optimal results with respect to the onset detection problem.

In this context, we also want to recall an important aspect concerning the result given by the quality measure. The diffusion process without a directional preference gives a higher precision than the diffusion process with a directional preference but this comes at the cost of a decrease in recall. The result for the process with (only) vertical diffusion improves not just the precision but also the recall with respect to the original data. This example shows that we always have to consider the quantities precision and recall together, compare Section 2.3.

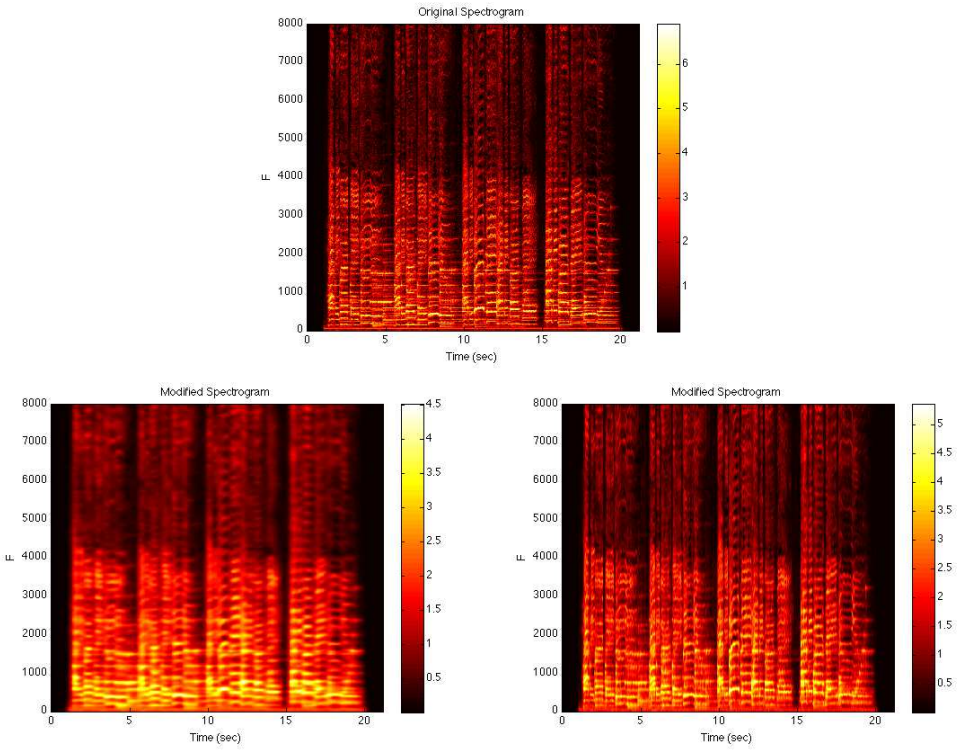

Figure 8.1: Spectrograms of "Happy Birthday" (number 4 in Table A.2). **Top:** Original spectrogram. **Bottom left:** Spectrogram after diffusion with EED and $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$, $t = 20$ where no direction is preferred. **Bottom right:** Spectrogram after diffusion with the same setting, but directional preference for vertical diffusion.

Table 8.1: Comparison of precision, recall and F-measure for EED with and without directional preference. Edge-enhancing diffusion was performed with $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$ and $t = 20$. The values correspond to the processing depicted in Figure 8.1.

Setting	Precision	Recall	F-Measure
Original	0.383	0.692	0.493
No preference	0.571	0.462	0.511
Vertical preference	0.455	0.769	0.571

Increasing the number of iterations. As we have seen in Table 8.1, diffusion with vertical preference gives improved results in comparison to unbiased anisotropic diffusion. We proceed by fixing the diffusion preference to vertical direction and investigate how increasing the number of iterations influences the result. Figure 8.2 shows again the "Happy Birthday" testfile. The spectrograms are processed by edge-enhancing diffusion with different numbers of iterations. The F-measure given in Table 8.2 suggests that the result for 100 iterations is an outlier and the more iterations we perform the more the final result is enhanced. Considering all the quantities of the quality measures, we see that the high precision and the high F-measure for 200 iterations comes with a low recall which is not desired. For 20 iterations we observe an improved result since we have an increase in precision as well as in recall in comparison to the original data.

Table 8.2: Comparison of precision, recall and F-measure for EED with different numbers of iterations and directional preference for vertical structures where $\lambda = 1$, $\sigma = 0.5$ and $\tau = 0.25$. The values correspond to the processing depicted in Figure 8.2.

Iterations	Precision	Recall	F-Measure
0	0.383	0.692	0.493
20	0.455	0.769	0.571
100	0.442	0.731	0.551
200	0.487	0.731	0.585

8.1.2 Coherence-Enhancing Diffusion

Vertical vs. no directional preference. In the context of coherence-enhancing diffusion, we begin again by investigating the impact of diffusion only in vertical direction in comparison to unbiased anisotropic diffusion. Figure 8.3 shows the spectrograms of the testfile "guitar3" after processing the original spectrogram with coherence-enhancing diffusion with and without directional preference. In general, we have to note that the spectrogram of this testfile is very dense. Thus, investigating the resulting spectrograms carefully is necessary to spot the direction of the diffusion process. Nevertheless, having a look at the frequency contributions (in yellow color) slightly below 5000, we see that they are also blurred in horizontal

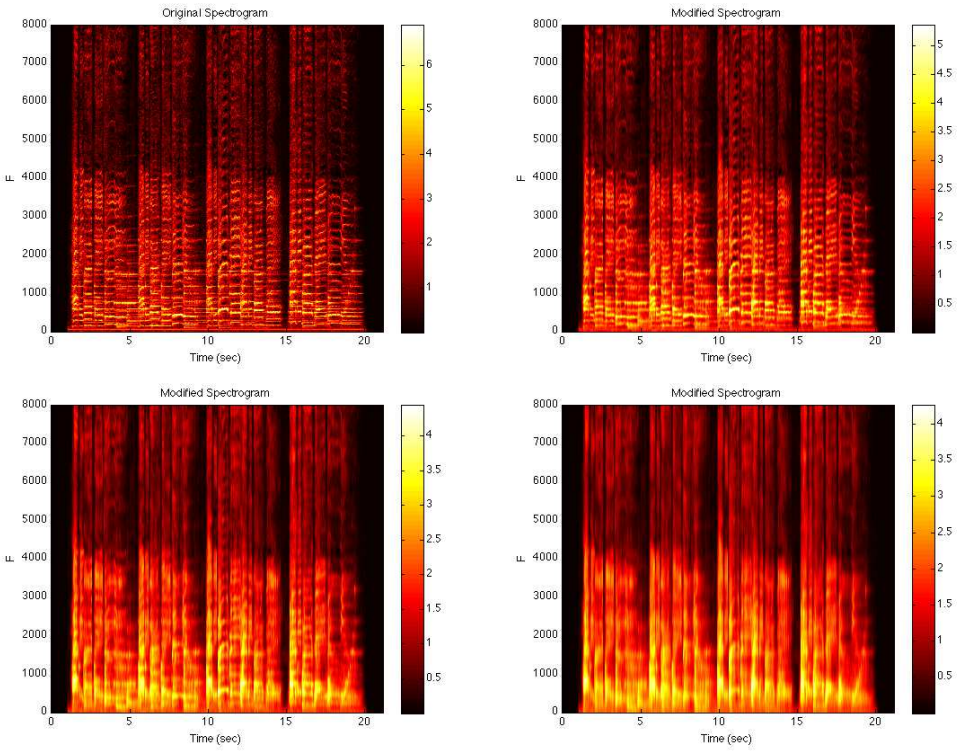

Figure 8.2: Spectrograms of "Happy Birthday" (number 4 in Table A.2). All diffusion processes are performed with a preference for vertical structures. **Top left:** Original spectrogram. **Top right:** Spectrogram after diffusion with EED and $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$ and $t = 20$. **Bottom left:** $t = 100$. **Bottom right:** $t = 200$.

direction in the spectrogram on the bottom left where these contributions are not horizontally extended in the version on the bottom right. Alternatively, we can investigate the frequencies that are above 6000. In the original spectrogram there are thin vertical structures. These are not only preserved but also enhanced by the diffusion algorithm with preference in vertical direction where these structures are blurred away for the diffusion process that performs unbiased anisotropic diffusion, compare the spectrograms on the bottom right and bottom left in Figure 8.3, respectively. Table 8.3 also shows that the setting with vertical preference is superior to the setting without directional preference.

Increasing the number of iterations. We proceed by investigating how an increase in the number of iterations influences the result of the coherence-enhancing diffusion where we fix the process to only diffuse in vertical direction, see Figure 8.4. As already mentioned before, the spectrogram of the "guitar3" testfile is very dense. Performing a large number of iterations introduces an even higher density which does not enhance the overall quality of the resulting spectrogram, see Table 8.4. For 500 or 1000 iterations, the recall is lower than without any processing which shows that the task of onset detection is more difficult for these spectrograms. In contrast, we notice that for 100 iterations, the result is slightly improved. In this

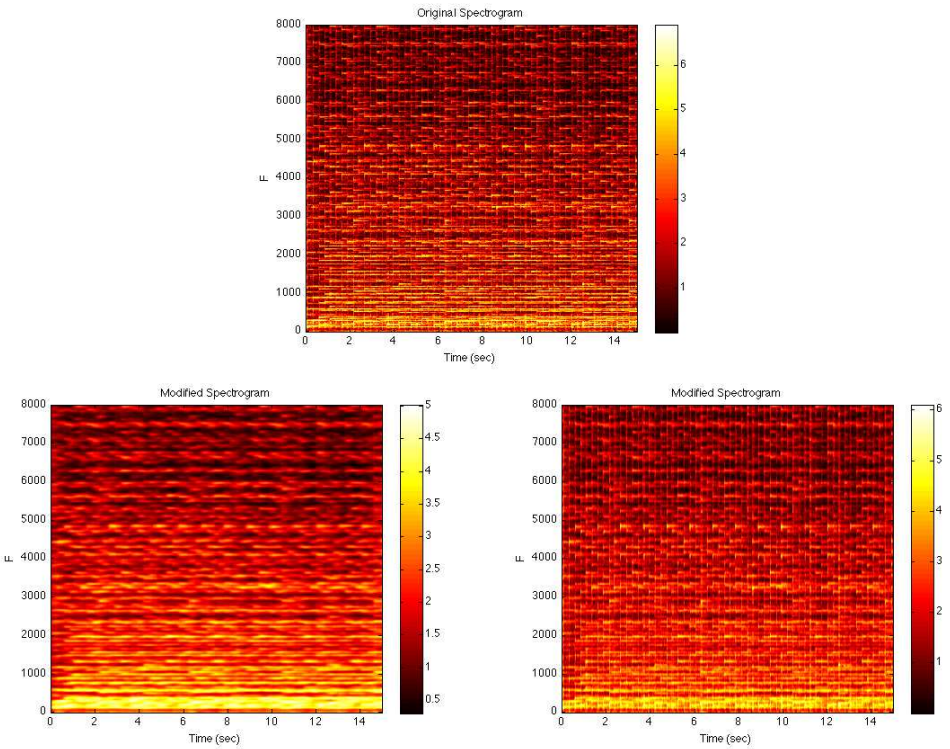

Figure 8.3: Spectrograms of "guitar3" (number 15 in Table A.2). **Top:** Original spectrogram. **Bottom left:** Spectrogram after diffusion with CED and $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 0.1$, $\tau = 0.25$, $t = 100$ where no direction is preferred. **Bottom right:** Spectrogram after diffusion with the same setting, but directional preference for vertical diffusion.

case, information about the structure is still preserved where for 500 or 1000 iterations, this information is blurred completely, compare e.g. the bottom of the corresponding spectrograms (in bright yellow).

Table 8.3: Comparison of precision, recall and F-measure for CED with and without directional preference where $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 0.1$, $\tau = 0.25$ and $t = 100$. The values correspond to the processing depicted in Figure 8.3.

Setting	Precision	Recall	F-Measure
Original	1.000	0.828	0.906
No preference	1.000	0.034	0.067
Vertical preference	1.000	0.845	0.916

Table 8.4: Comparison of precision, recall and F-measure for CED ($C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 0.1$ and $\tau = 0.25$) with vertical preference and different numbers of iterations. The values correspond to the processing depicted in Figure 8.4.

Iterations	Precision	Recall	F-Measure
0	1.000	0.828	0.906
100	1.000	0.845	0.916
500	1.000	0.810	0.895
1000	1.000	0.810	0.895

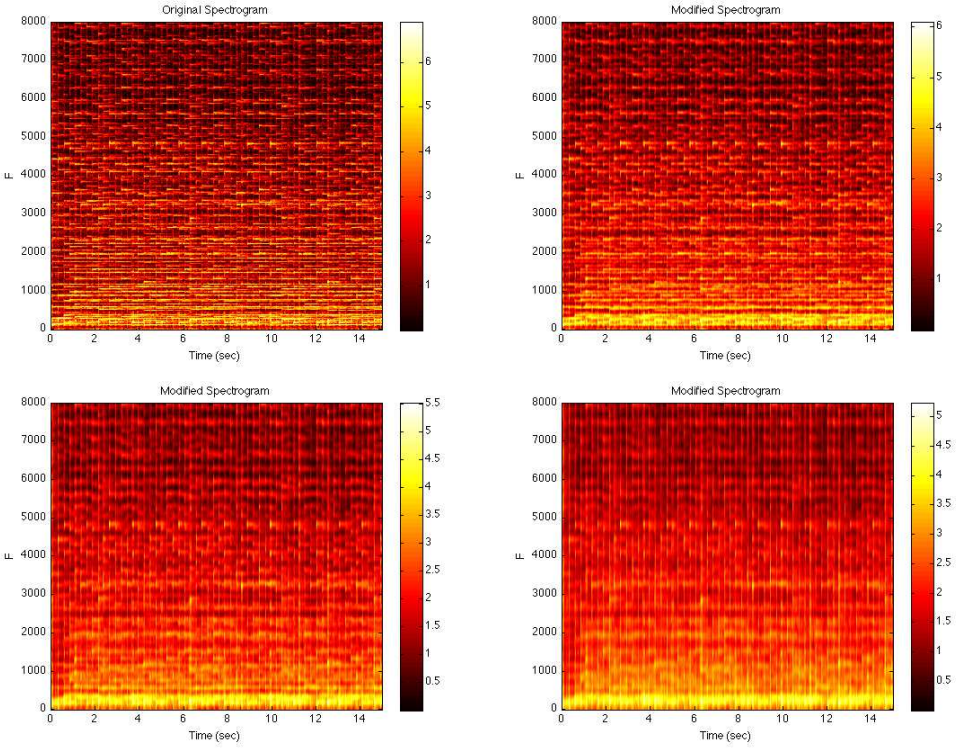

Figure 8.4: Spectrograms of "guitar3" (number 15 in Table A.2). All diffusion processes are performed with a preference for vertical structures. **Top left:** Original spectrogram. **Top right:** Spectrogram after diffusion with CED and $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 0.1$, $\tau = 0.25$, $t = 100$. **Bottom left:** $t = 500$. **Bottom right:** $t = 1000$.

8.1.3 Comparison of EED and CED

We have seen in the previous sections that edge-enhancing diffusion and coherence-enhancing diffusion behave similarly for onset detection: first, a process with directional preference is superior to a process without directional preference and second, the more a process diffuses, the worse gets the result. Like we did in Chapter 7, we compare EED and CED head to head while using similar parameter settings. Please remember that α in the CED algorithm is sort of a damping of the diffusion process. To allow a similar behavior of both diffusion algorithms for the same number of iterations, we have to set $\alpha = 1$ for coherence-enhancing diffusion.

The original spectrogram of the testfile Op. 100 No. 2 by Friedrich Burgmüller is given on the top of Figure 8.5. Below this, the spectrograms of the data processed with edge-enhancing diffusion and coherence-enhancing diffusion are given side by side. In the middle row, we compare EED and CED where no directional preference is used. Visually, the two results look very similar, but – according to the quality measure given in Table 8.5 – the spectrogram computed by the application of EED beats the one computed with CED. Please note that both diffusion algorithms without directional preference worsen the result compared to the original, unprocessed data which coincides with previous findings. The results of the diffusion processes with directional preference for vertical structures is given on the bottom of Figure 8.5. Again, the visual judgement reveals no differences between the spectrogram on the bottom left and the one on the bottom right. In this case, also the quality measure given in Table 8.5 shows that the two algorithms perform similarly.

But why do the two diffusion processes act in a similar fashion? To answer this, we refer to Section 7.1.3. The explanation already given there for chord recognition also holds for onset detection.

Table 8.5: Comparison between EED with $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$, $t = 50$ and CED using $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 1$, $\tau = 0.25$, $t = 50$. The table corresponds to Figure 8.5.

Method	Preference	Precision	Recall	F-Measure
Original	–	0.979	0.676	0.800
EED	no	0.941	0.235	0.376
CED	no	0.800	0.176	0.289
EED	yes	1.000	0.691	0.817
CED	yes	1.000	0.691	0.817

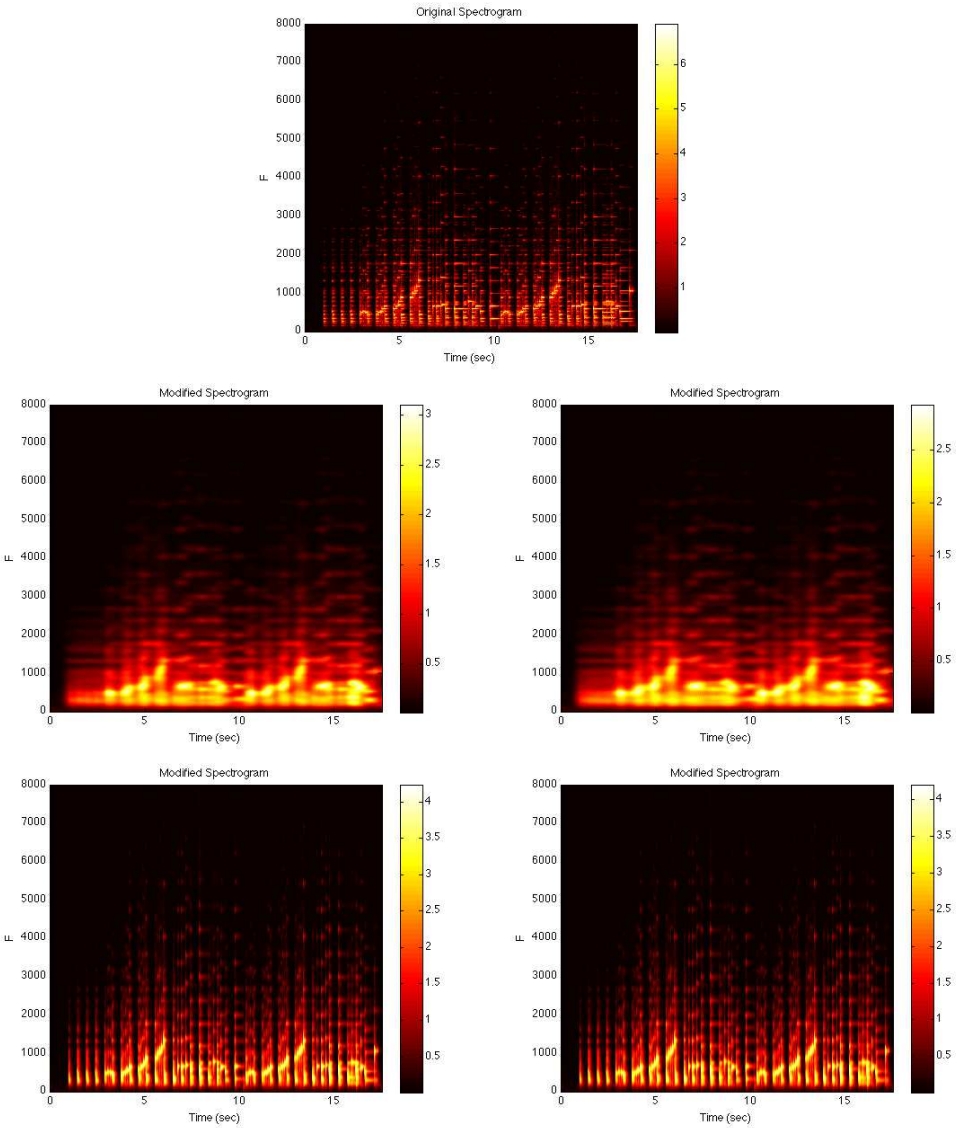

Figure 8.5: Comparison between EED and CED on the testfile Op. 100, No. 2 by Friedrich Burgmüller (number 1 in Table A.2). **Top:** Original spectrogram. **Middle left:** Edge-enhancing diffusion with $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$, $t = 50$ and no preference for vertical diffusion. **Bottom left:** Same setting with preference for vertical diffusion. **Middle right:** Coherence-enhancing diffusion with $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 1$, $\tau = 0.25$, $t = 50$ and no preference for vertical diffusion. **Bottom right:** Same parameters, but directional preference for vertical diffusion turned on.

8.2 Morphology

Similarly to Section 7.2, we start by investigating the basic building blocks of mathematical morphology, i.e. dilation, erosion, closing and opening, with a rectangle as (flat) structuring element before proceeding with a parabola (which is a nonflat structuring element) used in parabolic dilation, parabolic erosion and parabolic skeletonization. All methods are performed on the spectrogram of "rock1" given in Figure 8.6.

Please note that we omit the usage of a disk since we have already seen in Section 7.2 that a (flat) disk is not well suited for the processing of spectrograms.

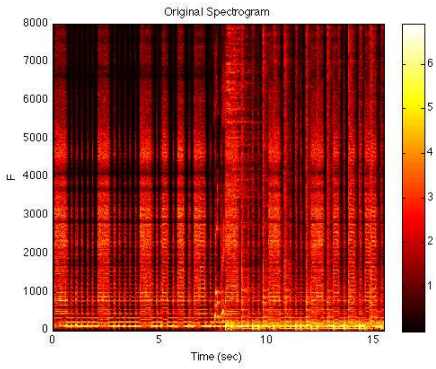

Figure 8.6: Spectrogram of "rock1" testfile (number 20 in Table A.2).

8.2.1 Dilation and Erosion

In the top row of Figure 8.7, the results of a dilation (top left) and an erosion (top right) with a rectangle of height 10 and width 1 are depicted. We notice that the dilation process introduces a high amount of blurring in vertical direction in comparison to the original spectrogram, see also Table 8.6. The problem is that such an enhancement might also give a higher impact to contributions that are no real onsets (e.g. noise). In contrast, erosion with the same structuring element reduces small contributions such that in general only strong contributions survive. For onset detection this is advantageous, since only "true" onsets may be present. On the other hand, actual onsets with a low contribution in the spectrogram may be eroded away. Considering Table 8.6, the application of erosion improves the result more than the dilation operation. The precision measure is slightly lower when erosion is applied, but the recall measure is significantly higher which leads to a result with improved overall quality. Please also note that the recall for erosion is not only higher in comparison to dilation but also to the original data.

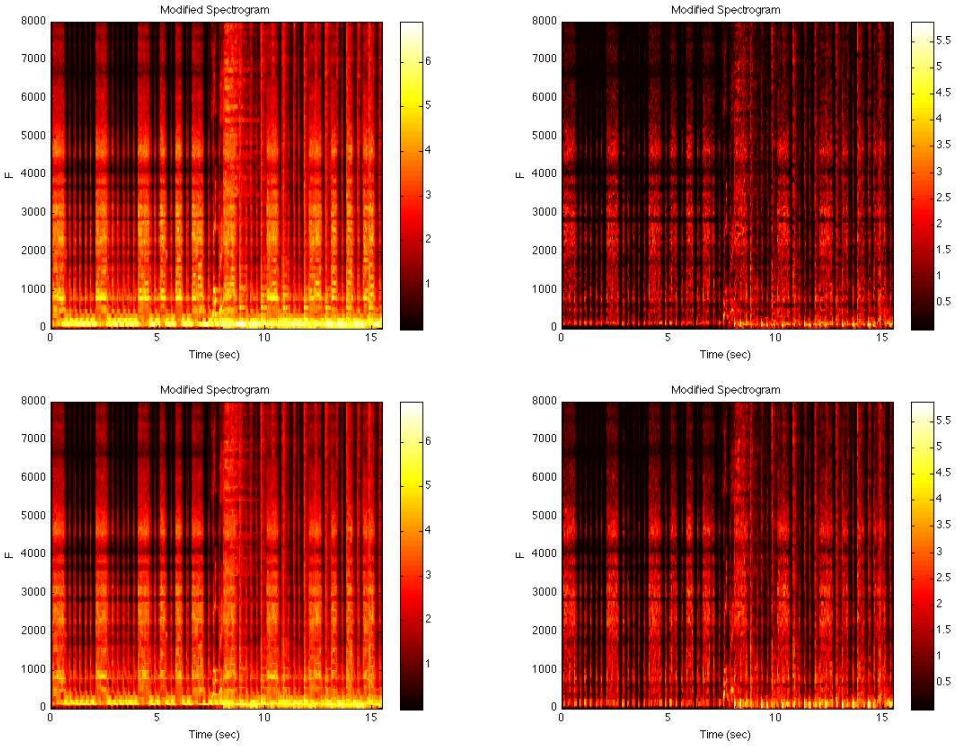

Figure 8.7: Morphological operations on the spectrogram of “rock1” with a rectangle of height 10 and width 1 as structuring element. **Top left:** Dilation. **Top right:** Erosion. **Bottom left:** Closing. **Bottom right:** Opening.

Table 8.6: Morphological operations on the spectrogram of “rock1” with a rectangle of height 10 and width 1 as structuring element. See also Figure 8.7.

Method	Precision	Recall	F-measure
Original	1.000	0.758	0.862
Dilation	1.000	0.710	0.830
Erosion	0.962	0.806	0.877
Closing	0.977	0.694	0.811
Opening	0.940	0.758	0.839

8.2.2 Opening and Closing

In this section, we are concerned with the operations opening and closing. The closing operation on the testfile "rock1" is depicted in Figure 8.7 on the bottom left and the opening operation on the bottom right. Please note that the results on the bottom can be achieved by the application of the opposite morphological operation on the spectrograms in the top row. In other words, the spectrogram on the bottom left of Figure 8.7 can be computed by applying an erosion operation with the same structuring element on the spectrogram depicted on the top left (showing a dilation). Consequently, the spectrogram on the bottom right can be computed by applying a dilation to the spectrogram on the top right.

Let us look at the closing in Figure 8.7 first. The blurring introduced by the dilation (see top left spectrogram) is reduced. Nevertheless, the dilation operation already introduced a high amount of blurring which cannot be compensated by the followed erosion. The opening operation is depicted in Figure 8.7 on the bottom right. Here we find that contributions that are almost eroded away (compare top right spectrogram) are restored by the dilation step. This also re-introduces noisy structures that were reduced by the previous erosion operation.

In Chapter 7 we have seen that the result of dilation was enhanced by applying erosion afterwards and erosion was enhanced by the successive application of dilation. In other words, the closing operation gave enhancements compared to pure dilation and the opening operation improved the overall result compared to pure erosion. For the task of onset detection, this does not seem to hold. The closing operation worsens the result in comparison to the dilation and the opening in comparison to the erosion, see Table 8.6.

Please also note that only erosion was able to enhance the overall quality of the testfile.

8.2.3 Skeletonization

As we did in Section 7.2.3, we start by briefly looking at parabolic dilation and parabolic erosion before proceeding to parabolic skeletonization. In addition, we compare the parabolic skeletonization approach as introduced in 6.2 with the binary skeletonization available in MATLAB.

Figure 8.8 shows parabolic dilation and parabolic erosion of "rock1". In case of dense spectrograms like in this example, parabolic dilation blurs even more than dilation with a rectangle as structuring element. With parabolic erosion, we observe the opposite. Parabolic erosion performs – in comparison to erosion with a rectangle as structuring element – stronger thinning of the vertical structures.

As already mentioned, MATLAB provides a skeletonization function (`bwmorph`) which only works on binary images. Thus, the first step is to convert the grayscale image to a binary version. We use again the function `graythresh` to compute a threshold for `im2bw` which converts an image to a binary version of it. Details can also be found in Section 7.2.3. Figure 8.9 shows the original spectrogram and its binary version as well as the skeletonization computed out of the binary version of the original spectrogram. If we take a look at the spectrograms on the bottom, we see that the skeleton consists of a lot of thin, crisscross lines. This makes it difficult to clearly determine onsets.

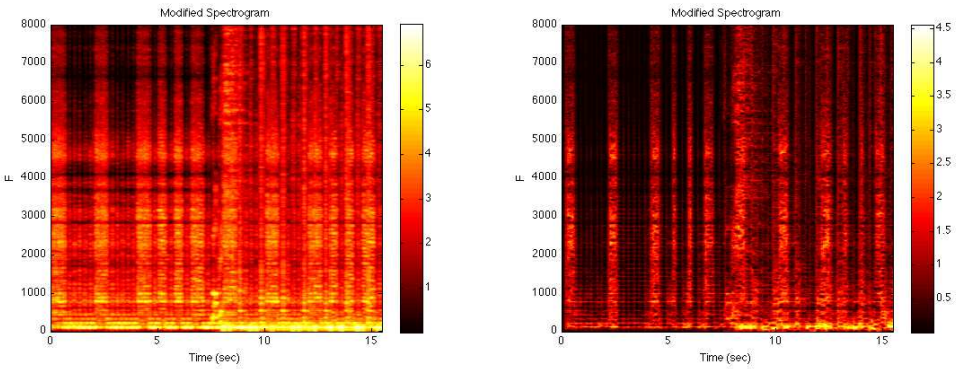

Figure 8.8: Parabolic morphological operations on the spectrogram of "rock1". **Left:** Parabolic dilation with $s_x = 0.1$ and $s_y = 0.1$. **Right:** Parabolic erosion with $s_x = 0.1$ and $s_y = 0.1$.

Table 8.7: Comparison of skeletonization algorithms on "rock1". The first two skeletonizations correspond to the `bwmorph` function that is built into MATLAB. The binary image for `bwmorph` was generated using `im2bw` with an automatically chosen threshold realized by `graythresh`. Last three are parabolic skeletonizations with a threshold of 0.01 for different values for s_x and s_y.

Method	s_x	s_y	Precision	Recall	F-measure
Original	–	–	1.000	0.758	0.862
MATLAB	–	–	0.525	0.339	0.412
MATLAB + color	–	–	0.595	0.355	0.444
Parabolic	0.1	0.1	0.939	0.742	0.829
Parabolic	1	0.1	0.935	0.694	0.796
Parabolic	0.1	1	0.918	0.726	0.811

Figure 8.10 compares binary skeletonization as implemented in MATLAB with the parabolic skeletonization proposed in Section 6.2. We notice immediately that both approaches are very different in their outcome. Parabolic skeletonization keeps the overall structure intact while thinning it at the same time where the skeletonization implemented in MATLAB gives very thin lines that are not related to the original structure in any way. Like for chord recognition, the differences between the versions where $s_x = s_y = 0.1$, $s_x = 1, s_y = 0.1$ and $s_x = 0.1, s_y = 1$ are very small. Looking carefully, we can see a blurring in horizontal direction for the spectrogram on the bottom left and an enhancement of vertical structures for the spectrogram on the bottom right. It seems that less blurring is involved for $s_x = s_y = 0.1$ leading to the conclusion that this version – at least by judging visually – is the best out of the three parabolic skeletonization examples. Table 8.7 supports this impression.

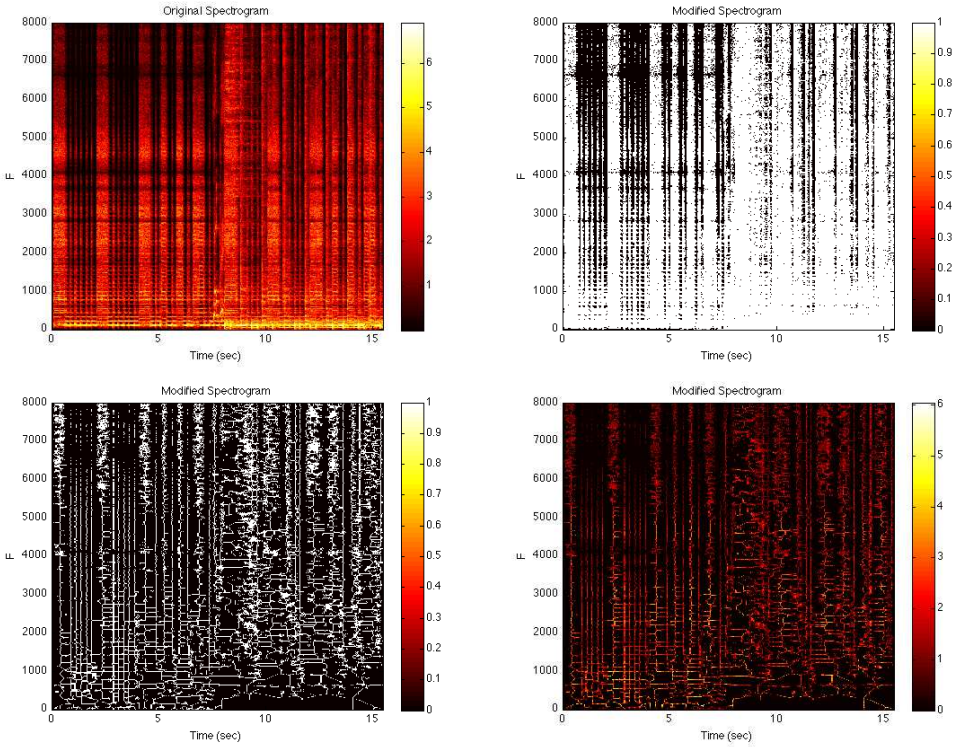

Figure 8.9: Skeletonization of "rock1". **Top left:** Original spectrogram. **Top right:** Binary image of the original spectrogram generated using the built-in MATLAB function `im2bw` with an automatically chosen threshold realized by the built-in function `graythresh`. **Bottom left:** Skeleton computed with the MATLAB function `bwmorph` for skeletonization using the binary version of the original spectrogram. **Bottom right:** Same skeleton with color information of the original spectrogram at skeleton points.

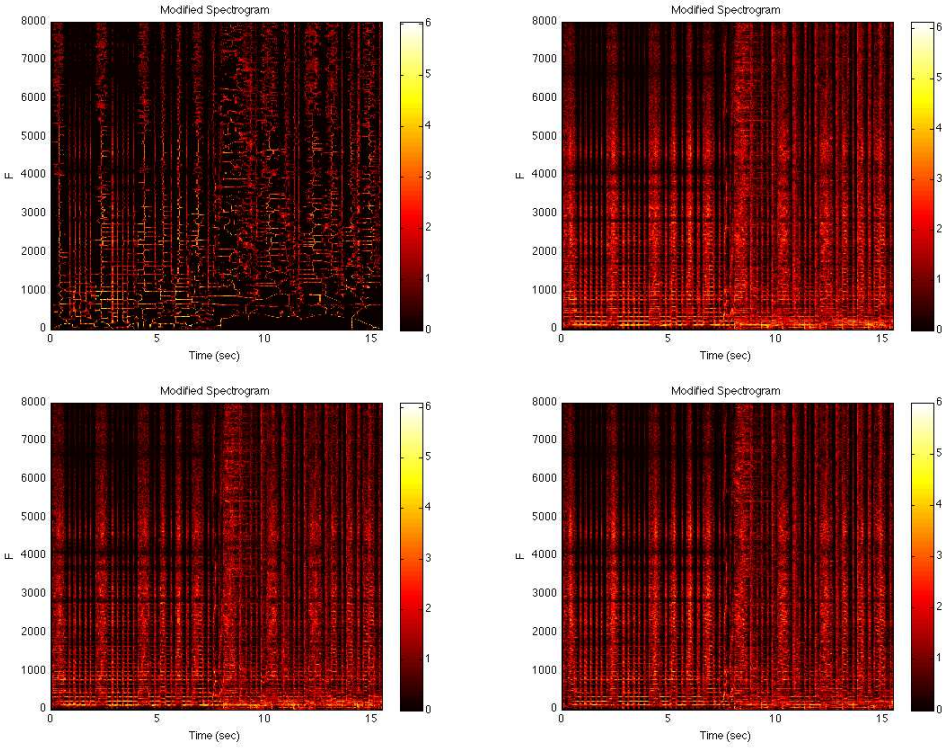

Figure 8.10: Different skeletonization approaches on "rock1". **Top left:** Skeleton computed with `bwmorph` and additional color information. **Top right:** Parabolic skeletonization with $s_x = 0.1$, $s_y = 0.1$ and a threshold of 0.01. **Bottom left:** Same, but $s_x = 1$, $s_y = 0.1$. **Bottom right:** Same, but $s_x = 0.1$, $s_y = 1$.

8.3 NL-Means

As we have seen in Section 7.3, the modifications introduced in Sections 5.2 and 6.3 introduce assumptions that influence the outcome of the NL-means algorithm. Please remember that we stick to the term *NL-means* to refer to an algorithm incorporating improvements from 5.2 and 6.3 but not the modified weighting function from (5.9), whereas *fast NL-means* stands for the algorithm that additionally uses this modified weighting function.

In the case of NL-means and chord recognition, we had to be careful choosing what we are inpainting. As a consequence, we had to make the search window and the similarity window large enough to ensure this. Please note that NL-means has a high memory consumption, especially for large window sizes. This might lead to the exhaustion of system capacities very quickly. Experiments for onset detection with the same window sizes as for chord recognition have led to a poorer outcome. Let us show the difference on two examples. NL-means with a 21×21 search window, a similarity window of size 3×21, $\lambda = 1$ and nonadaptive distance weighting as well as fast NL-means with $\eta_1 = 0.5$ and $\eta_2 = 1.5$ were used where all other parameters remained untouched. NL-means gives on Op. 100 No. 2 by Friedrich Burgmüller $(P, R, F) = (1.000, 0.632, 0.775)$ for precision, recall and F-measure where fast NL-means gives $(P, R, F) = (1.000, 0.647, 0.786)$. For the testfile "classic3", NL-means ends with $(P, R, F) = (0.438, 0.583, 0.500)$ and fast NL-means with $(P, R, F) = (0.526, 0.833, 0.645)$. Comparing those results with the corresponding ones in Tables 8.14 and 8.25, respectively, we notice that a larger window size is not suitable for this task. As a consequence, we stick to a smaller search window of size 11×11 and a smaller similarity window (11×11 or 3×11, respectively). As a side effect, the smaller window sizes save some system resources.

8.3.1 Improved NL-means

We start with the comparison between NL-means with and without adaptive weighting (see Section 5.2.3) for different parameters λ.

No directional preference. Figure 8.11 shows on the left the output spectrograms of NL-means without adaptive weighting and on the right the spectrograms that are computed with an adaptively chosen distance weight λ. First of all we may see that the spectrograms where NL-means was applied without adaptively chosen λ are a little more blurred than the spectrograms where NL-means with adaptive weighting was used. However, the changes between the left spectrogram and its corresponding version on the right are minimal. As a general trend we observe that the higher λ is chosen, the more blurring is involved in the spectrograms leading to poorer results with the increase of λ. Table 8.8 verifies these observations. Furthermore, the table shows that for $\lambda = 5$ and $\lambda = 10$, a larger difference between the version without and with adaptive weighting is present. Let us exemplarily study the two spectrograms for $\lambda = 5$. More precisely, we have a look at the "vertical peak" at approximately 7.5 seconds. Especially at the frequencies around 3500 to 4000 we may see that NL-means without an adaptive distance weight λ blurs very much in contrast to the version with adaptive weighting

Table 8.8: Comparison of choosing λ adaptively or not for NL-means. A 11×11 search window and a 11×11 similarity window is used. This table corresponds to Figure 8.11.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.917	0.805	0.857
1	no	0.971	0.829	0.895
1	yes	0.917	0.805	0.857
5	no	1.042	0.610	0.769
5	yes	1.037	0.683	0.824
10	no	0.520	0.317	0.394
10	yes	0.591	0.317	0.413

that preserves the vertical structure quite well. Similar investigations can be made e.g. at approximately 10.5 seconds for the frequencies around 2500 to 3000.

Except for $\lambda = 1$ without adaptive weighting, the result of NL-means is not enhanced in comparison to the original data.

Vertical directional preference. Let us proceed by exploring whether it helps to use a rectangular similarity window instead of a quadratic one. In Figure 8.12 we have again the spectrograms of "guitar2", but we use now a similarity window of size 3×11 instead of 11×11. On the left side, the spectrograms for ascending λ and nonadaptive weighting are shown from top to bottom where on the right side the spectrograms for the same values of λ but with adaptive weighting are depicted. Visually, hardly any differences can be spotted comparing the spectrograms on the left and on the right, but the F-measure values in Table 8.9 give a significant change between nonadaptive and adaptive weighting for $\lambda = 5$ and $\lambda = 10$.

Please note that in Section 7.3 we have given a reason that a rectangular search window is not well suited which is also true for this task. Of course, it might be that the usage of a rectangular search window improves the outcome of the algorithm, but this highly depends on the testfile that is considered.

Table 8.9: Comparison of choosing λ adaptively or not for NL-means. A 11×11 search window and a 3×11 similarity window is used. This table corresponds to Figure 8.12.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.917	0.805	0.857
1	no	1.032	0.780	0.889
1	yes	0.970	0.780	0.865
5	no	0.963	0.634	0.765
5	yes	1.036	0.707	0.841
10	no	0.588	0.488	0.533
10	yes	0.767	0.561	0.648

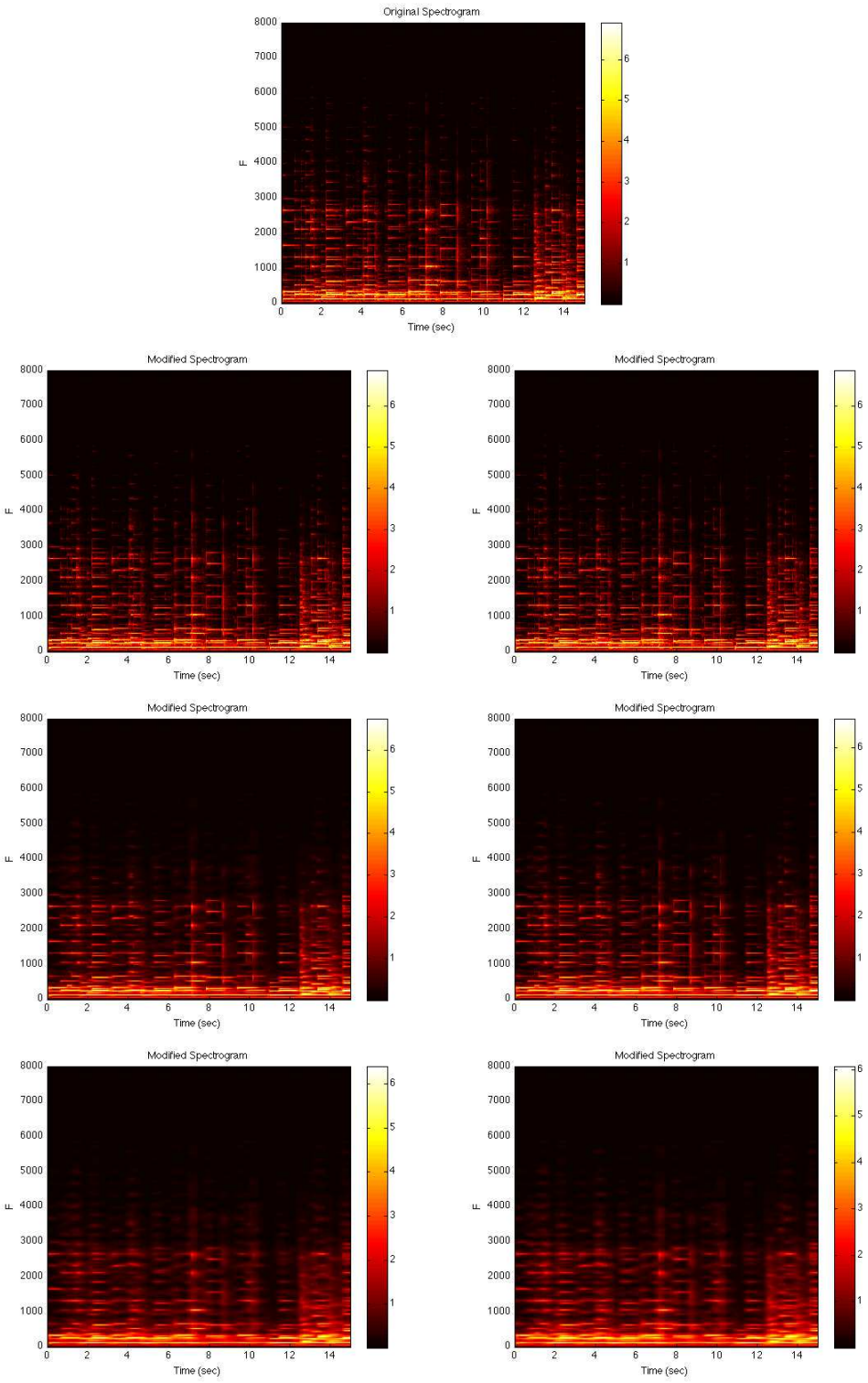

Figure 8.11: Comparison of choosing λ adaptively or not for NL-means using the testfile "guitar2" (number 14 in Table A.2). All spectrograms were computed using a 11×11 search window and a 11×11 similarity window. **Top:** Original spectrogram. **Left:** λ is not chosen adaptively. **Right:** λ is chosen adaptively. **Middle to bottom:** $\lambda = 1, 5, 10$.

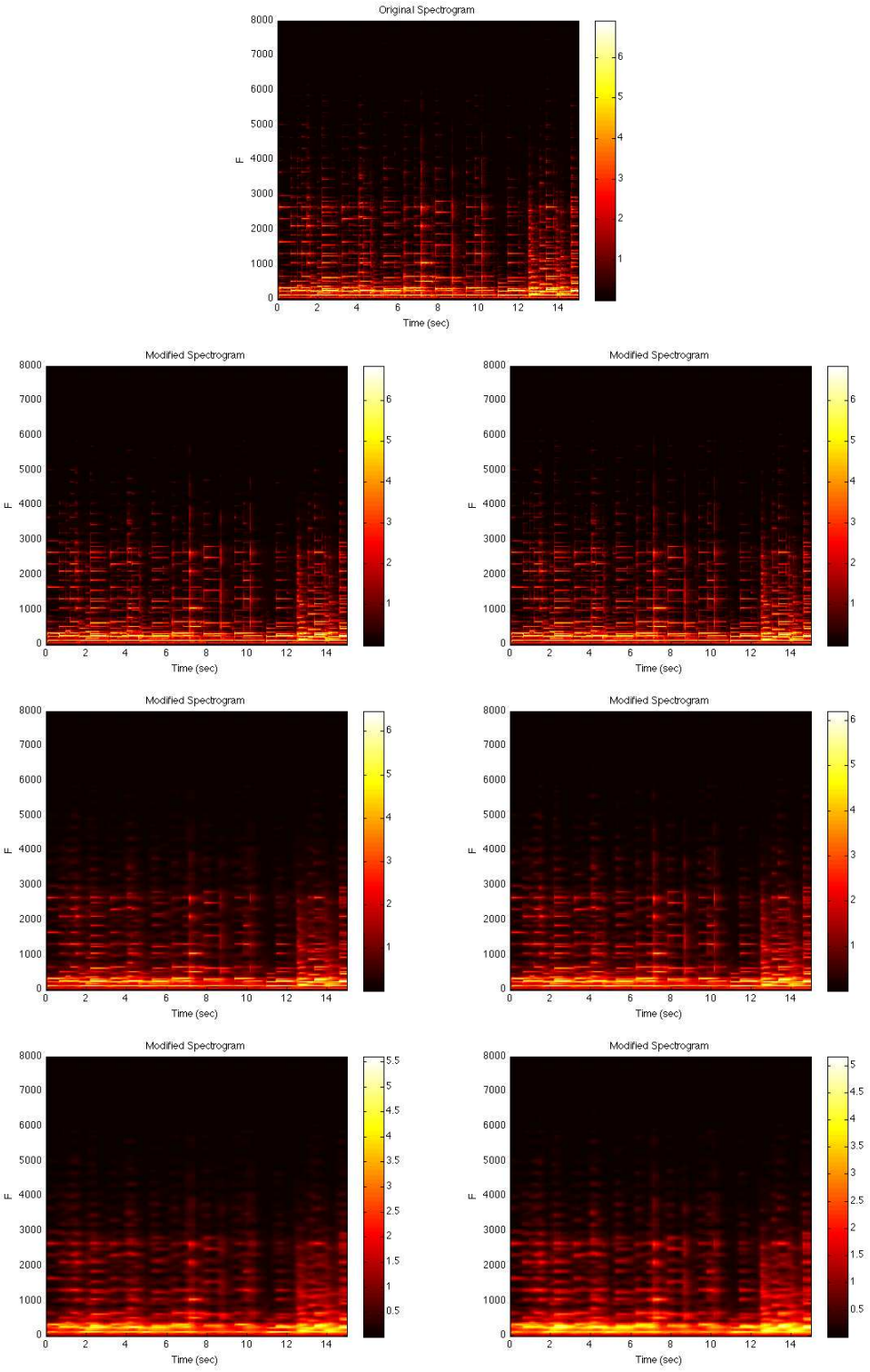

Figure 8.12: Same as in Figure 8.11, but with a 11×11 search window and a 3×11 similarity window.

8.3.2 Fast NL-means

We now turn our attention to the fast NL-means algorithm that uses the modified weighting function as given in equation (5.9). In this context, we compare again the results for fast NL-means with and without directional preference for the similarity window. In addition to that, we use two different parameter sets for η_1 and η_2 where the first one is $\eta_1 = 0.9, \eta_2 = 1.1$ and the second one is $\eta_1 = 0.5, \eta_2 = 1.5$.

No directional preference. Let us start with a search window of size 11×11 and an equally sized similarity window. Figure 8.13 compares the impact of using the adaptive weighting function for different values of λ. Furthermore, $\eta_1 = 0.9$ and $\eta_2 = 1.1$ were used for this example. The results of the same setting but with $\eta_1 = 0.5$ and $\eta_2 = 1.5$ can be found in Figure 8.14. Since visually these figures do not differ significantly, let us focus on the results given by the quality measure. These results can be found in Table 8.10 and Table 8.11, respectively.

First of all, we note that the outcome of the fast NL-means algorithm without and with adaptive weighting slightly differs. Furthermore, both parameter sets for η_1 and η_2 give the same trend concerning the increase of λ, i.e. the higher λ is chosen the worse is the result. This fact can be seen on the bottom row of Figures 8.13 and 8.14 where the spectrograms are blurred very much. Another regularity seems to be that the version with adaptive weighting is superior in comparison to the version without adaptive weighting for higher λ.

Table 8.10: Comparison of choosing λ adaptively or not for fast NL-means with a 11×11 search window, a 11×11 similarity window, $\eta_1 = 0.9$ and $\eta_2 = 1.1$. This table corresponds to Figure 8.13.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.917	0.805	0.857
1	no	0.943	0.805	0.868
1	yes	0.919	0.829	0.872
5	no	1.000	0.659	0.794
5	yes	0.966	0.683	0.800
10	no	0.625	0.488	0.548
10	yes	0.767	0.561	0.648

Vertical directional preference. We now turn to the case that the similarity window has a rectangular shape, i.e. 3×11.

Figure 8.15 contains the spectrograms computed by the application of the fast NL-means algorithm using $\eta_1 = 0.9$ and $\eta_2 = 1.1$. Figure 8.16 gives the results for fast NL-means with $\eta_1 = 0.5$ and $\eta_2 = 1.5$. The first thing we notice in both figures is that hardly any blurring is involved in the result. Nevertheless, with a higher λ also the blurring increases.

Table 8.11: Comparison of choosing λ adaptively or not for fast NL-means. A 11×11 search window, a 11×11 similarity window and $\eta_1 = 0.5$, $\eta_2 = 1.5$ is used. This table corresponds to Figure 8.14.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.917	0.805	0.857
1	no	0.944	0.829	0.883
1	yes	0.919	0.829	0.872
5	no	1.000	0.659	0.794
5	yes	1.000	0.683	0.812
10	no	0.429	0.293	0.348
10	yes	0.483	0.341	0.400

Let us assume that $\eta_1 = 0.9$ and $\eta_2 = 1.1$ are chosen. In Table 8.12, we see that $\lambda = 5$ without adaptive weighting gives the highest F-measure. This outcome is not expected, since the computation with $\lambda = 1$ results in less blurring in the spectrograms compared to the other values of λ. The fact that the computation with $\lambda = 5$ gives together with a higher precision also a lower recall compared to the original data supports our intuition that this result should be taken with care. In contrast, for $\lambda = 1$ we have an increase in precision together with the same recall as for the original data. If $\eta_1 = 0.5$ and $\eta_2 = 1.5$ are used, the result for $\lambda = 1$ without adaptive weighting is superior to all other results, compare Table 8.13.

But why is there hardly any blurring involved in this setting? This circumstance results from the usage of the rectangular similarity window which limits the amount of inpainting in horizontal direction and thus reduces blurring.

Table 8.12: Comparison of choosing λ adaptively or not for fast NL-means with $\eta_1 = 0.9$ and $\eta_2 = 1.1$ where a 11×11 search window and a 3×11 similarity window is used. This table corresponds to Figure 8.15.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.917	0.805	0.857
1	no	0.971	0.805	0.880
1	yes	0.943	0.805	0.868
5	no	1.032	0.780	0.889
5	yes	1.000	0.780	0.877
10	no	1.000	0.780	0.877
10	yes	1.000	0.780	0.877

Table 8.13: Comparison of choosing λ adaptively or not for fast NL-means with $\eta_1 = 0.5$ and $\eta_2 = 1.5$. Furthermore, a 11×11 search window and a 3×11 similarity window is used. This table corresponds to Figure 8.16.

λ	Adaptive	Precision	Recall	F-measure
Original	–	0.917	0.805	0.857
1	no	1.000	0.805	0.892
1	yes	0.971	0.805	0.880
5	no	0.969	0.756	0.849
5	yes	1.033	0.756	0.873
10	no	0.889	0.585	0.706
10	yes	0.929	0.634	0.754

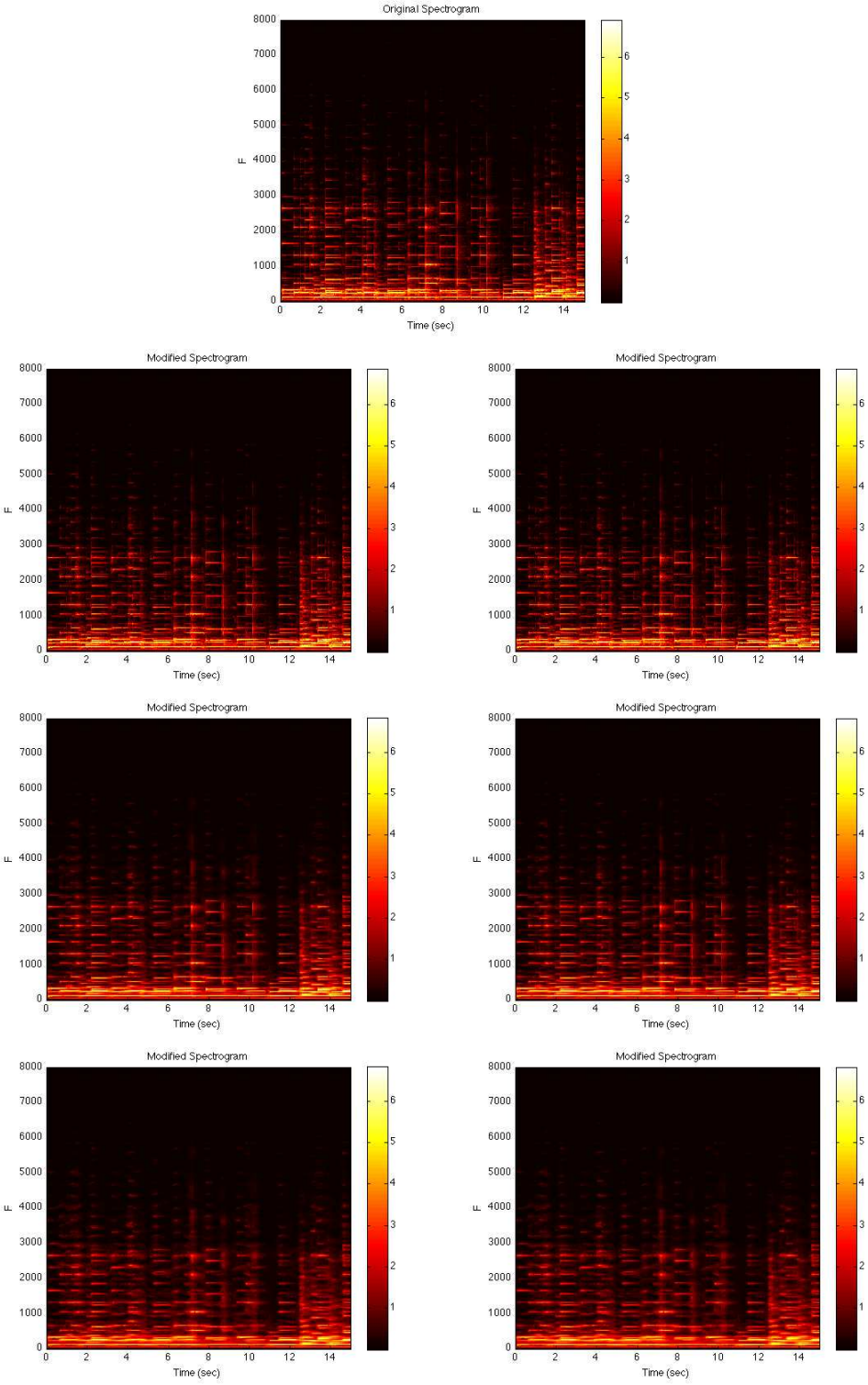

Figure 8.13: Comparison of choosing λ adaptively or not for fast NL-means using the testfile "guitar2" (number 14 in Table A.2). All spectrograms were computed using a 11×11 search window, a 11×11 similarity window, $\eta_1 = 0.9$ and $\eta_2 = 1.1$. **Top:** Original spectrogram. **Left:** λ is not chosen adaptively. **Right:** λ is chosen adaptively. **Middle to bottom:** $\lambda = 1, 5, 10$.

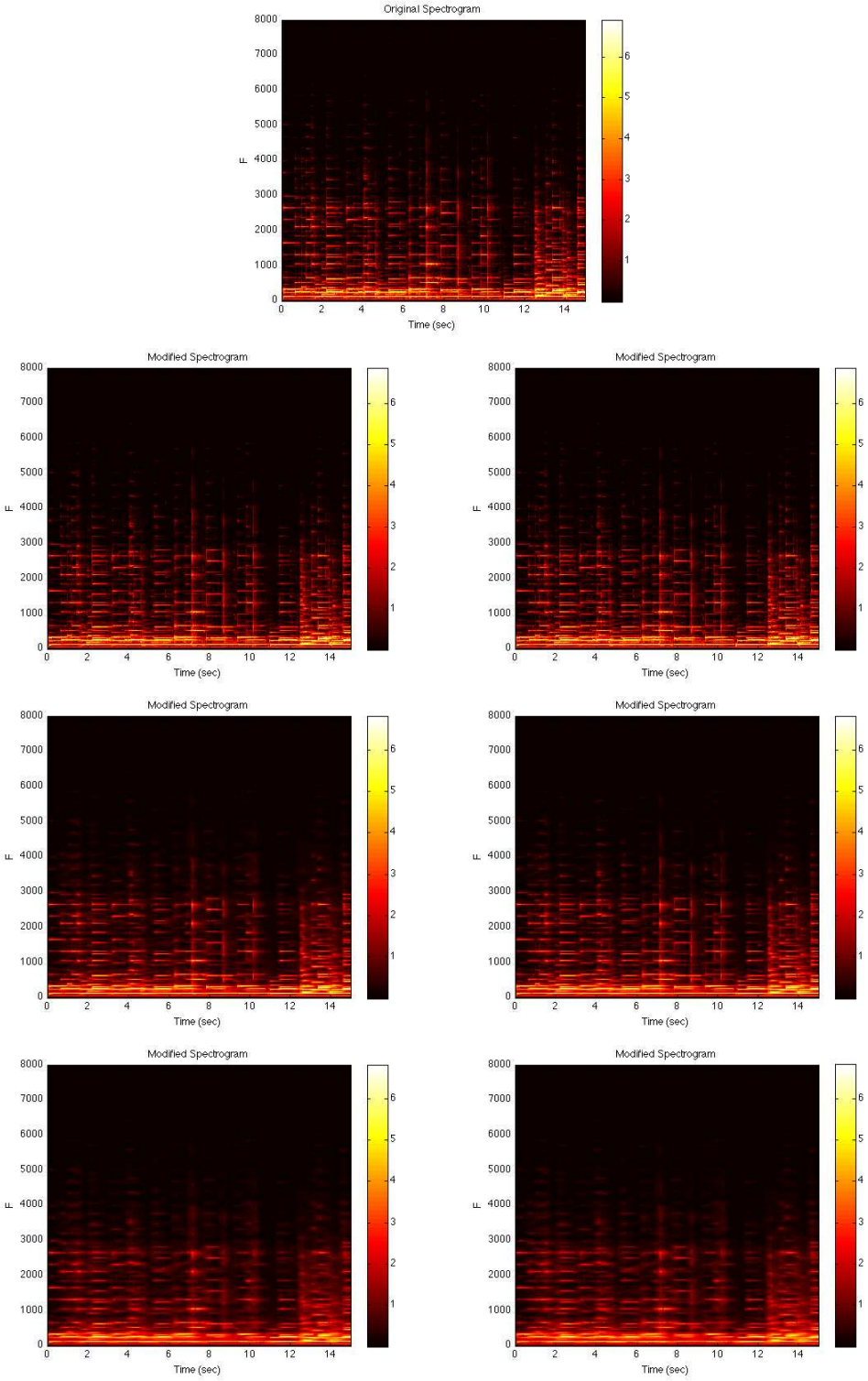

Figure 8.14: Same as in Figure 8.13, but with $\eta_1 = 0.5$ and $\eta_2 = 1.5$.

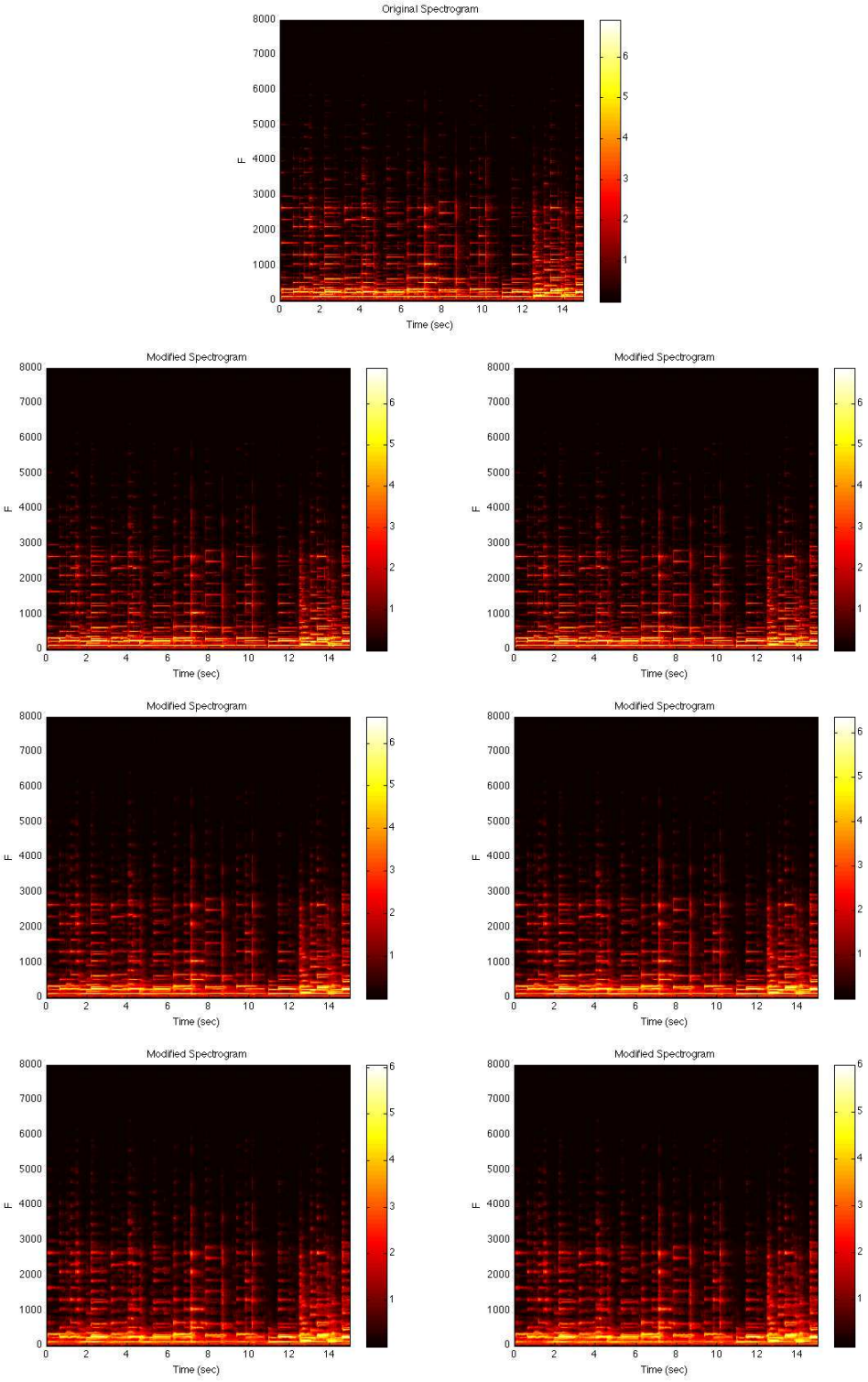

Figure 8.15: Comparison of choosing λ adaptively or not for fast NL-means using the testfile "guitar2" (number 14 in Table A.1). All spectrograms were computed using a 11×11 search window, a 3×11 similarity window, $\eta_1 = 0.9$ and $\eta_2 = 1.1$. **Top:** Original spectrogram. **Left:** λ is not chosen adaptively. **Right:** λ is chosen adaptively. **Middle to bottom:** $\lambda = 1, 5, 10$.

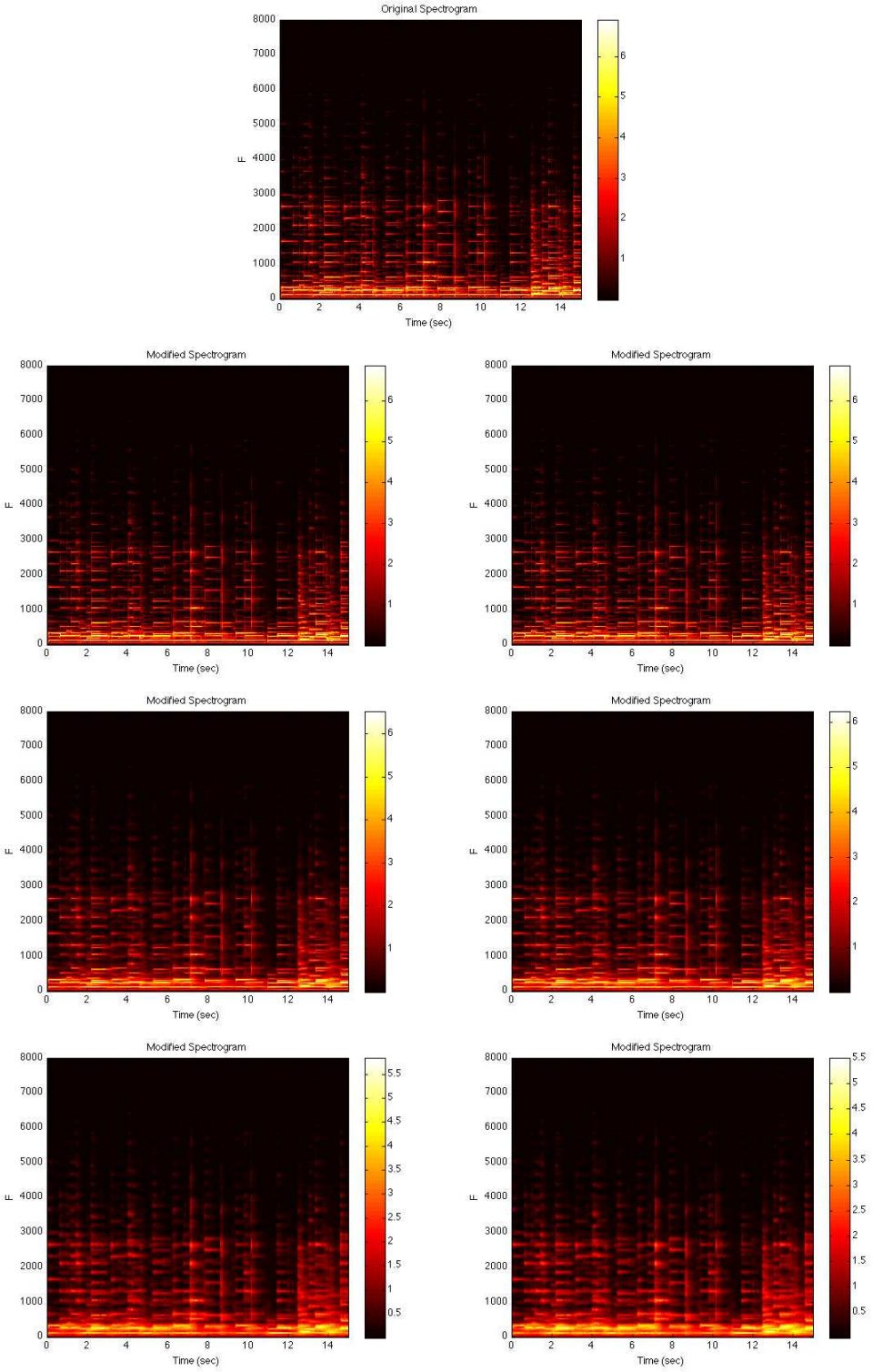

Figure 8.16: Same as in Figure 8.15, but with $\eta_1 = 0.5$ and $\eta_2 = 1.5$.

8.4 Discussion and Summary

This section concludes the experiments for the onset detection problem. We do so by using a standard parameter set for EED, CED, skeletonization, NL-means and fast NL-means on all the testfiles and compare for each file how well the algorithms perform. Like in Section 7.4, we are interested in a reasonable starting point concerning the choice of parameters for the algorithms under investigation, since in real-world examples, usually no ground truth is available. As it is for the chord recognition problem, standardized parameter sets are crucial for an automated application of the methods. Since the impact of the algorithms is visually demonstrated in the previous sections, we omit to give the spectrograms like we did in Section 7.4.

For the computations, the following parameter sets are applied and preference for vertical structures is always used except for skeletonization.

- EED:
  $\lambda = 1$, $\sigma = 0.5$, $\tau = 0.25$, $t = 20$.
- CED:
  $C = 1$, $\sigma = 0.5$, $\rho = 5$, $\alpha = 0.01$, $\tau = 0.25$, $t = 200$.
- Skeletonization:
  $s_x = 0.1$, $s_y = 0.1$, threshold = 0.01.
- NL-means:
  11×11 search window, 3×11 similarity window, $\lambda = 1$, no adaptive weighting.
- Fast NL-means:
  11×11 search window, 3×11 similarity window, $\eta_1 = 0.5$, $\eta_2 = 1.5$, $\lambda = 1$, no adaptive weighting.

Performance comparison across all approaches. Concerning diffusion, we should keep in mind the following. The spectrograms of the testfiles for onset detection investigated in this thesis are in most cases very dense. Thus, we have to be very careful when choosing the diffusion time; choosing it too large might lead to unwanted results.

The outcome of the processing by the algorithms investigated in this thesis is given for each testfile in Tables 8.14 to 8.36. As a general trend, we note that the class of diffusion processes beats the class of NL-means algorithms on the testfiles, compare e.g. Tables 8.19 to 8.22. Also, skeletonization outperforms both NL-means approaches on several testfiles, e.g. on the testfile “jazz3” (see Table 8.30). Please note that for “jazz3”, skeletonization also beats diffusion. Furthermore, the NL-means algorithm that does not use the weighting function as given in equation (5.9) gives better results in comparison to the fast NL-means. Interestingly, NL-means and fast NL-means beat diffusion on testfiles where mainly vocals are present, compare Tables 8.17 and 8.18, respectively. In contrast, diffusion gives a slightly higher F-measure for “classic2” even though this file also contains vocals, compare Table 8.24. The

reason might be that the singing starts at approximately half of the audio excerpt where the first half contains only piano. In addition, the low frequencies on the bottom of the spectrogram are already blurred throughout the unprocessed testfile, compare also the corresponding original spectrogram in Figure A.4.

So far, we compared the algorithms among each other. Considering the overall improvement, i.e. whether it is beneficial to apply one of the algorithms on the testfiles, we notice that for some examples no improvement according to precision, recall and F-measure is achieved at all, see Tables 8.16, 8.20, 8.21, 8.24, 8.25, 8.29, 8.31 and 8.33.

Runtime considerations. Even if runtime is not in the main focus of this thesis, we don't want to leave it out of consideration. Table 8.37 gives the runtime of the algorithms investigated in this thesis on the testfile "cello1" (number 9 in Table A.2) using the respective parameter sets as they are given before. In general, the runtime of the algorithms is lower than for the chord recognition problem, compare Table 7.28, because we are using less iterations for diffusion or use smaller window sizes in the NL-means algorithms.

Combination of different filters. A natural idea (like in the case of chord recognition) is to combine two of the methods to further enhance the spectrograms. Since diffusion and NL-means surpass skeletonization on the problems, we combine EED and fast NL-means. Applied on its own, edge-enhancing diffusion needs in general few iteration steps to enhance a spectrogram for the task of onset detection. In combination with fast NL-means, the small changes of the diffusion process are somehow compensated by the non-local inpainting which gives a result very similar to the one for fast NL-means alone. Thus, we modify the parameter set proposed earlier in this section by increasing the number of iterations for the EED algorithm to $t = 200$ while leaving the rest of the EED parameter set and the parameters for fast NL-means untouched. Table 8.38 shows the results of fast NL-means as preprocessing or postprocessing operation for edge-enhancing diffusion. Figure 8.17 contains the corresponding spectrograms. For the sake of readability, we use *FNLM* instead of fast NL-means in Table 8.38.

The first thing we notice when looking at the spectrograms in Figure 8.17 is that the application of EED, be it either as preprocessing, postprocessing or alone, introduces some amount of blurring. Secondly, the spectrograms on the middle left and on the bottom look quite similar. Thus, it is difficult to say by visual comparison why EED is superior to the versions where fast NL-means is used once as preprocessing and once as postprocessing step. Except for pure EED, the results are quite close based on the values for precision, recall and F-measure given in Table 8.38.

Final remark. Onset detection seems to be insensitive to noise that appears locally which is the type of noise we try to reduce by image processing methods. Thus, we are only able to slightly enhance the spectrograms.

Table 8.14: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for Op. 100 No. 2 by Friedrich Burgmüller (file number 1 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.979	0.676	0.800
EED	1.000	0.691	0.817
CED	0.979	0.676	0.800
Skeletonization	0.979	0.676	0.800
NL-means	1.000	0.662	0.796
Fast NL-means	1.000	0.662	0.796

Table 8.15: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for Beethoven's Fifth in a Bernstein interpretation (file number 2 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.552	0.296	0.386
EED	0.538	0.389	0.452
CED	0.500	0.352	0.413
Skeletonization	0.429	0.278	0.337
NL-means	0.565	0.241	0.338
Fast NL-means	0.524	0.204	0.293

Table 8.16: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for Hungarian Dance by Brahms (file number 3 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.983	0.731	0.838
EED	0.950	0.731	0.826
CED	0.966	0.731	0.832
Skeletonization	0.966	0.731	0.832
NL-means	0.964	0.692	0.806
Fast NL-means	0.964	0.692	0.806

Table 8.17: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “Happy Birthday” (file number 4 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.383	0.692	0.493
EED	0.455	0.769	0.571
CED	0.426	0.769	0.548
Skeletonization	0.306	0.577	0.400
NL-means	0.556	0.769	0.645
Fast NL-means	0.556	0.769	0.645

Table 8.18: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “In Frankrijk buiten de poorten” (file number 5 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.500	0.737	0.596
EED	0.517	0.789	0.625
CED	0.500	0.737	0.596
Skeletonization	0.483	0.737	0.583
NL-means	0.583	0.737	0.651
Fast NL-means	0.560	0.737	0.636

Table 8.19: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for Beethoven's Fifth as MIDI file (file number 6 in Table A.2).

Method	Precision	Recall	F-measure
Original	1.133	0.405	0.596
EED	1.161	0.429	0.626
CED	1.097	0.405	0.591
Skeletonization	0.903	0.333	0.487
NL-means	1.040	0.310	0.477
Fast NL-means	1.045	0.274	0.434

Table 8.20: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for String Quartet No. 2 by Alexander Borodin (file number 7 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.396	0.462	0.426
EED	0.346	0.462	0.396
CED	0.389	0.449	0.417
Skeletonization	0.372	0.449	0.407
NL-means	0.372	0.372	0.372
Fast NL-means	0.373	0.359	0.366

Table 8.21: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for Waltz No. 2 from Jazz Suite No. 2 by Dmitri Shostakovich (file number 8 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.971	0.878	0.922
EED	0.980	0.861	0.917
CED	0.970	0.852	0.907
Skeletonization	0.939	0.800	0.864
NL-means	0.961	0.852	0.903
Fast NL-means	0.970	0.835	0.897

Table 8.22: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “cello1” (file number 9 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.932	0.846	0.887
EED	0.984	0.938	0.961
CED	0.948	0.846	0.894
Skeletonization	0.930	0.815	0.869
NL-means	0.981	0.785	0.872
Fast NL-means	0.962	0.769	0.855

Table 8.23: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “clarinet1” (file number 10 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.170	0.211	0.188
EED	0.196	0.237	0.214
CED	0.174	0.211	0.190
Skeletonization	0.143	0.184	0.161
NL-means	0.205	0.237	0.220
Fast NL-means	0.196	0.237	0.214

Table 8.24: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “classic2” (file number 11 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.625	0.714	0.667
EED	0.623	0.673	0.647
CED	0.596	0.694	0.642
Skeletonization	0.630	0.694	0.660
NL-means	0.646	0.633	0.639
Fast NL-means	0.644	0.592	0.617

Table 8.25: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “classic3” (file number 12 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.588	0.833	0.690
EED	0.290	0.750	0.419
CED	0.435	0.833	0.571
Skeletonization	0.462	0.500	0.480
NL-means	0.556	0.833	0.667
Fast NL-means	0.556	0.833	0.667

Table 8.26: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “distguit1” (file number 13 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.833	0.750	0.789
EED	0.833	0.750	0.789
CED	0.842	0.800	0.821
Skeletonization	0.833	0.750	0.789
NL-means	0.824	0.700	0.757
Fast NL-means	0.824	0.700	0.757

Table 8.27: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “guitar2” (file number 14 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.917	0.805	0.857
EED	0.970	0.780	0.865
CED	0.919	0.829	0.872
Skeletonization	0.882	0.732	0.800
NL-means	1.032	0.780	0.889
Fast NL-means	1.000	0.805	0.892

Table 8.28: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “guitar3” (file number 15 in Table A.2).

Method	Precision	Recall	F-measure
Original	1.000	0.828	0.906
EED	1.000	0.810	0.895
CED	0.980	0.845	0.907
Skeletonization	1.000	0.793	0.885
NL-means	1.000	0.879	0.936
Fast NL-means	1.000	0.879	0.936

Table 8.29: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “jazz2” (file number 16 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.900	0.482	0.628
EED	0.897	0.464	0.612
CED	0.900	0.482	0.628
Skeletonization	0.862	0.446	0.588
NL-means	0.923	0.429	0.585
Fast NL-means	0.889	0.429	0.578

Table 8.30: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “jazz3” (file number 17 in Table A.2).

Method	Precision	Recall	F-measure
Original	1.026	0.639	0.788
EED	1.027	0.623	0.776
CED	1.026	0.639	0.788
Skeletonization	1.000	0.672	0.804
NL-means	0.955	0.344	0.506
Fast NL-means	0.963	0.426	0.591

Table 8.31: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “piano1” (file number 18 in Table A.2).

Method	Precision	Recall	F-measure
Original	1.000	1.000	1.000
EED	1.000	1.000	1.000
CED	1.000	1.000	1.000
Skeletonization	1.000	0.950	0.974
NL-means	1.000	1.000	1.000
Fast NL-means	1.000	0.950	0.974

Table 8.32: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “pop1” (file number 19 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.771	0.844	0.806
EED	0.818	0.844	0.831
CED	0.794	0.844	0.818
Skeletonization	0.688	0.688	0.687
NL-means	0.794	0.844	0.818
Fast NL-means	0.800	0.875	0.836

Table 8.33: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “rock1” (file number 20 in Table A.2).

Method	Precision	Recall	F-measure
Original	1.000	0.758	0.862
EED	1.000	0.726	0.841
CED	1.000	0.758	0.862
Skeletonization	0.939	0.742	0.829
NL-means	1.000	0.710	0.830
Fast NL-means	1.000	0.710	0.830

Table 8.34: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “sax1” (file number 21 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.500	0.800	0.615
EED	0.500	0.900	0.643
CED	0.471	0.800	0.593
Skeletonization	0.500	0.800	0.615
NL-means	0.500	0.800	0.615
Fast NL-means	0.588	1.000	0.741

Table 8.35: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for “trumpet1” (file number 22 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.976	0.683	0.804
EED	0.953	0.683	0.796
CED	0.956	0.717	0.819
Skeletonization	0.909	0.667	0.769
NL-means	0.976	0.683	0.804
Fast NL-means	0.953	0.683	0.796

Table 8.36: Comparison of EED, CED, skeletonization, NL-means and fast NL-means for "violin2" (file number 23 in Table A.2).

Method	Precision	Recall	F-measure
Original	0.909	0.506	0.650
EED	0.933	0.532	0.677
CED	0.932	0.519	0.667
Skeletonization	0.905	0.481	0.628
NL-means	1.000	0.481	0.650
Fast NL-means	1.000	0.481	0.650

Table 8.37: Comparison of algorithm runtime for "cello1".

Method	Runtime [s]
EED	0.693905
CED	19.238360
Skeletonization	104.406383
NL-means	229.284294
Fast NL-means	151.289294

Table 8.38: Fast NL-means as preprocessing or postprocessing step for diffusion. This table gives the corresponding values for precision, recall and F-measure for the spectrograms given in Figure 8.17.

Application chain	Precision	Recall	F-measure
Original	1.026	0.639	0.788
EED	1.028	0.607	0.763
FNLM	0.963	0.426	0.591
EED $\rightarrow$ FNLM	0.964	0.443	0.607
FNLM $\rightarrow$ EED	0.962	0.410	0.575

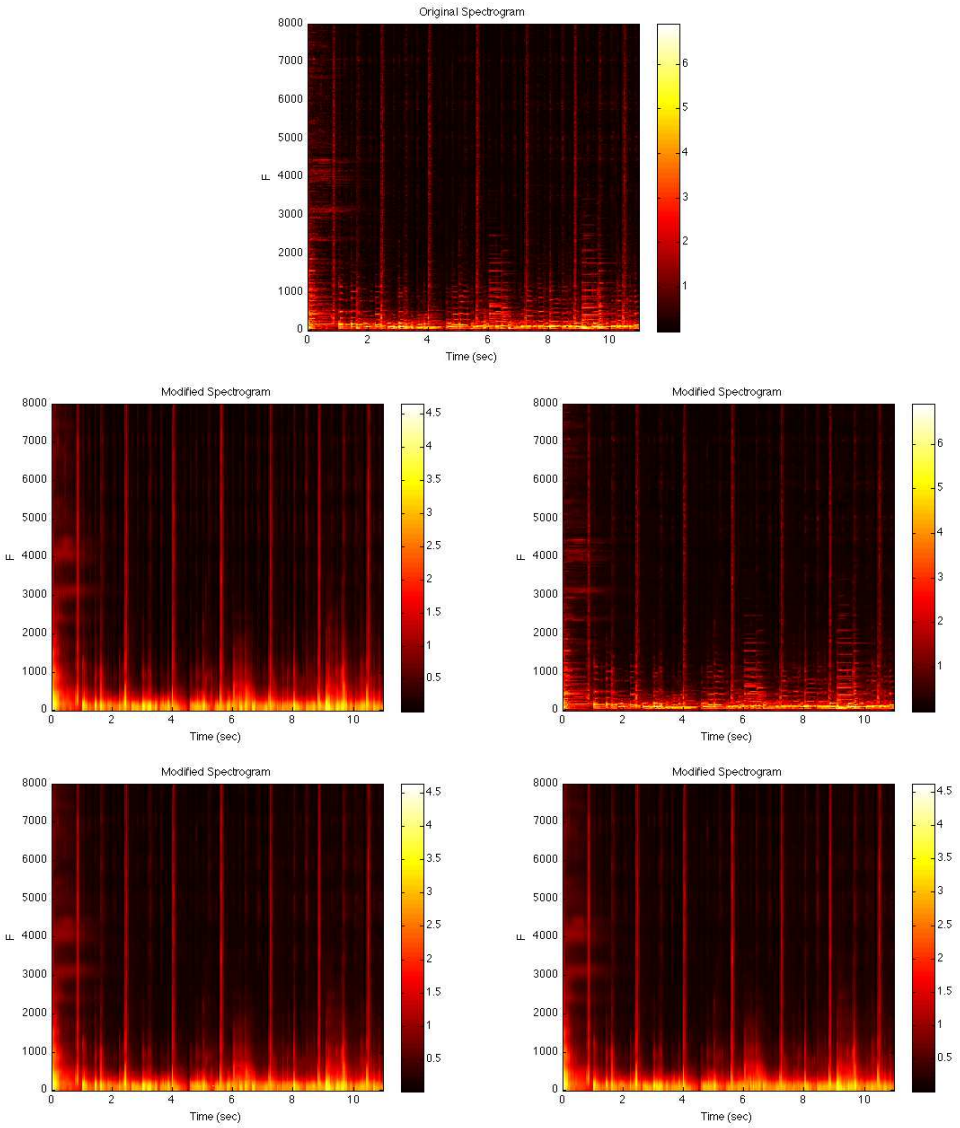

Figure 8.17: Fast NL-means as preprocessing or postprocessing step for diffusion. **Top:** Original spectrogram of "jazz3" (number 17 in Table A.2). **Middle left:** Spectrogram after applying EED with 200 iterations. **Middle right:** Fast NL-means with a 11×11 search window and a 3×11 similarity window, $\eta_1 = 0.5$, $\eta_2 = 1.5$, $\lambda = 1$ and no adaptive weighting. **Bottom left:** Application of a diffusion process followed by fast NL-means. **Bottom right:** Application of fast NL-means followed by a diffusion process.

Chapter 9

Conclusion and Future Work

If you're not prepared to be wrong, you'll never come up with anything original.
— Sir Ken Robinson

9.1 Conclusion

The goal of this thesis was to show whether methods from image processing can help to enhance spectrograms that are further used for feature extraction. We investigated two tasks, chord recognition and onset detection. The image processing methods considered in the scope of this thesis included diffusion processes (edge-enhancing diffusion and coherence-enhancing diffusion), mathematical morphology (dilation, erosion, closing, opening and skeletonization) as well as NL-means in various forms. Furthermore, we used the algorithms in their basic form which means that no knowledge of audio processing was included in the algorithmic design. In the case of NL-means, several approaches already available in the literature were implemented where their main purpose is to speed up the runtime of the NL-means algorithm that is computationally expensive. In addition to that, we proposed a new approach to skeletonization: parabolic skeletonization.

Modified versions that use a directional preference when processing a spectrogram were also implemented for all algorithms. This modification is important since chord recognition relies on horizontal structures and onset detection on vertical structures. For NL-means, this directional preference can be realized by changing the similarity window from a quadratic shape to a rectangular one. For diffusion processes, we introduced a modified diffusion tensor to make sure that a certain direction is preferred.

For chord recognition, image processing algorithms could significantly improve the spectrograms in most of the cases according to the quality measure under consideration. The task of onset detection is more difficult. As a consequence, only marginal improvements were achieved. For some of the testfiles, even none of the algorithms investigated here could improve the result. In fact, image processing methods usually try to repair noise that appears locally. Since onset detection seems to be insensitive to this kind of deterioration, image processing methods cannot fruitfully serve as a preprocessing step and the focus should lie in

improving the algorithms used for onset detection.

Usually no ground truth is available in case of chord recognition as well as for onset detection. Thus, we proposed for each image processing algorithm a corresponding parameter set that can be used as a guideline for the application on real-world data.

In our experiments, we have seen that mainly diffusion algorithms with preference for horizontal or vertical direction (depending on the problem) can enhance the spectrograms where algorithms without directional preference did in general not succeed to do so.

9.2 Future Work

The image processing algorithms investigated in this thesis were used in their basic form. Besides slight modifications to meet certain requirements in the experiments, no further knowledge is involved. Incorporating knowledge about music and audio processing into the algorithms might enhance the overall result and may lead to a smaller runtime due to assumptions or simplifications potentially introduced by this knowledge. But also knowledge from music theory could be helpful to adapt the behavior of algorithms, e.g. usual chord progression patterns for chord recognition to reduce the impact of progressions that are unlikely.

It is also interesting to investigate how various image processing algorithms perform when applied consecutively on the chord recognition or onset detection problem. Such processing chains were shortly touched in Chapters 7 and 8. However, a deep analysis on the impact of such chains would have been beyond the scope of this thesis. At least for the task of chord recognition, improvements can be expected by the consecutive application of two or more algorithms.

NL-means produces state-of-the-art results for image denoising tasks and many improvements have been contributed. To provide a wide variety of different image processing methods in the context of this thesis, only some of the approaches to improve NL-means were investigated [34, 23] and others were left out of consideration, e.g. [26]. While the algorithm proposed by Mahmoudi et al. [23] is one of the best NL-means speed-up strategies, the approach is somewhat heuristic. Orchard et al. [26] suggest to use the singular value decomposition (SVD) and to make use of its statistical properties. Their experiments showed that this method significantly improves the denoising effect due to a refined discrimination between similar and dissimilar pixel neighborhoods. It would be interesting to explore how this approach performs on spectrograms. At least for chord recognition, a changed similarity measure might give better inpainting results.

The algorithms presented in this thesis rely on several parameters. While we have fixed most of them, it might be possible to achieve better results by tweaking some parameters or by choosing them adaptively in a similar fashion like the locally adaptive weighting for NL-means. This task may also incorporate knowledge from audio processing to find better parameter settings.

Beyond image processing algorithms, the usage of PLP curves as introduced in [16] instead of novelty curves as onset detector might give advantages since a PLP curve is more robust to outliers and reveals musically meaningful information.

Appendix A

Testfiles

In this thesis, several examples where used in the context of chord recognition and onset detection. Table A.1 gives an overview of the files used for chord recognition and Table A.2 lists the files used for onset detection. The spectrograms of the testfiles for chord recognition are given in Figures A.1 and A.2 and the spectrograms of the testfiles for onset detection in Figures A.3 to A.6.

Table A.1: Testfiles for chord recognition.

Number	Title	Duration [s]
1	All My Loving	25
2	All You Need Is Love	30
3	And I Love Her	29
4	Baby's In Black	37
5	Back In The USSR	30
6	Cry Baby Cry	30
7	Helter Skelter	29
8	I Am The Walrus	30
9	I'm Only Sleeping	34
10	Let It Be	26
11	Savoy Truffle	29
12	Tell Me Why	29
13	You Can't Do That	29

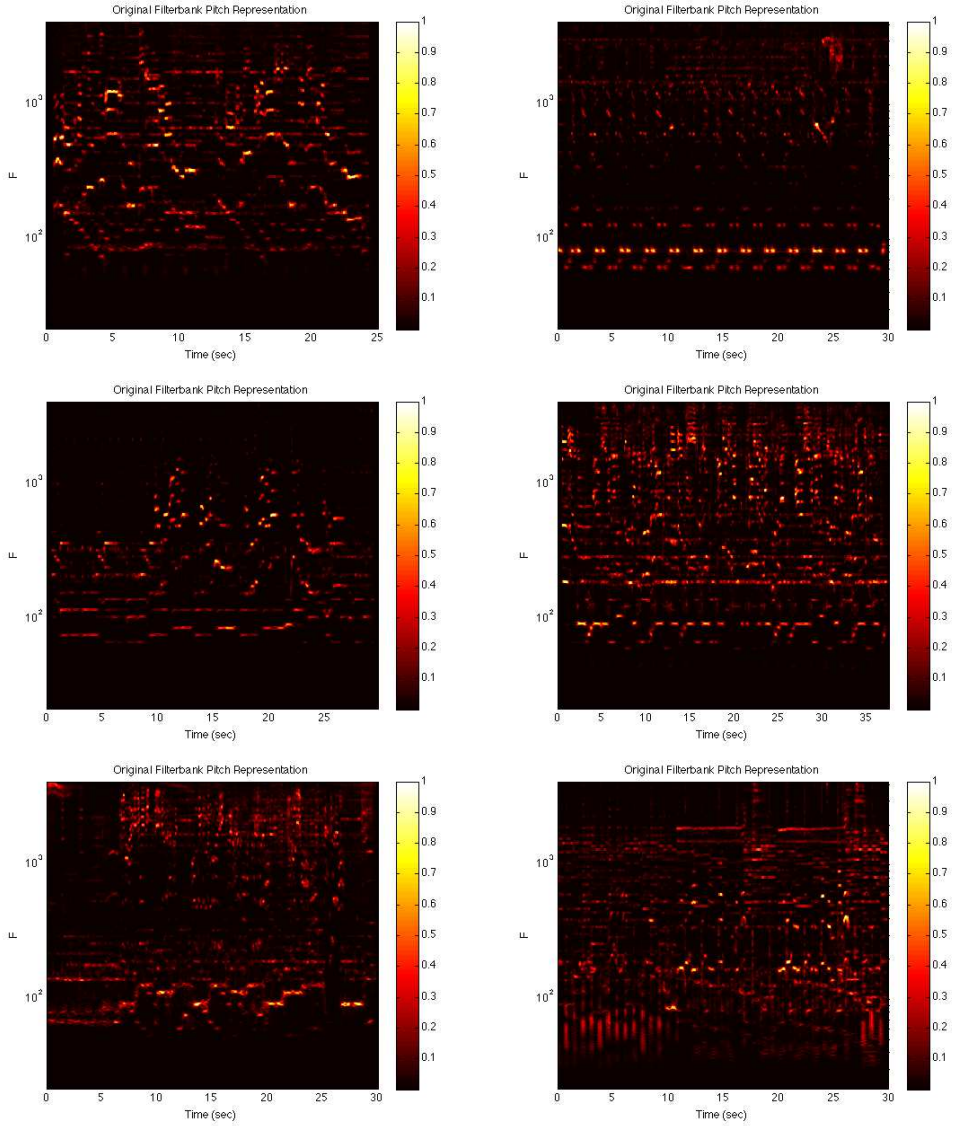

Figure A.1: Spectrograms of testfiles for chord recognition. **Left to right, top to bottom:** Testfiles 1 to 6, compare Table A.1.

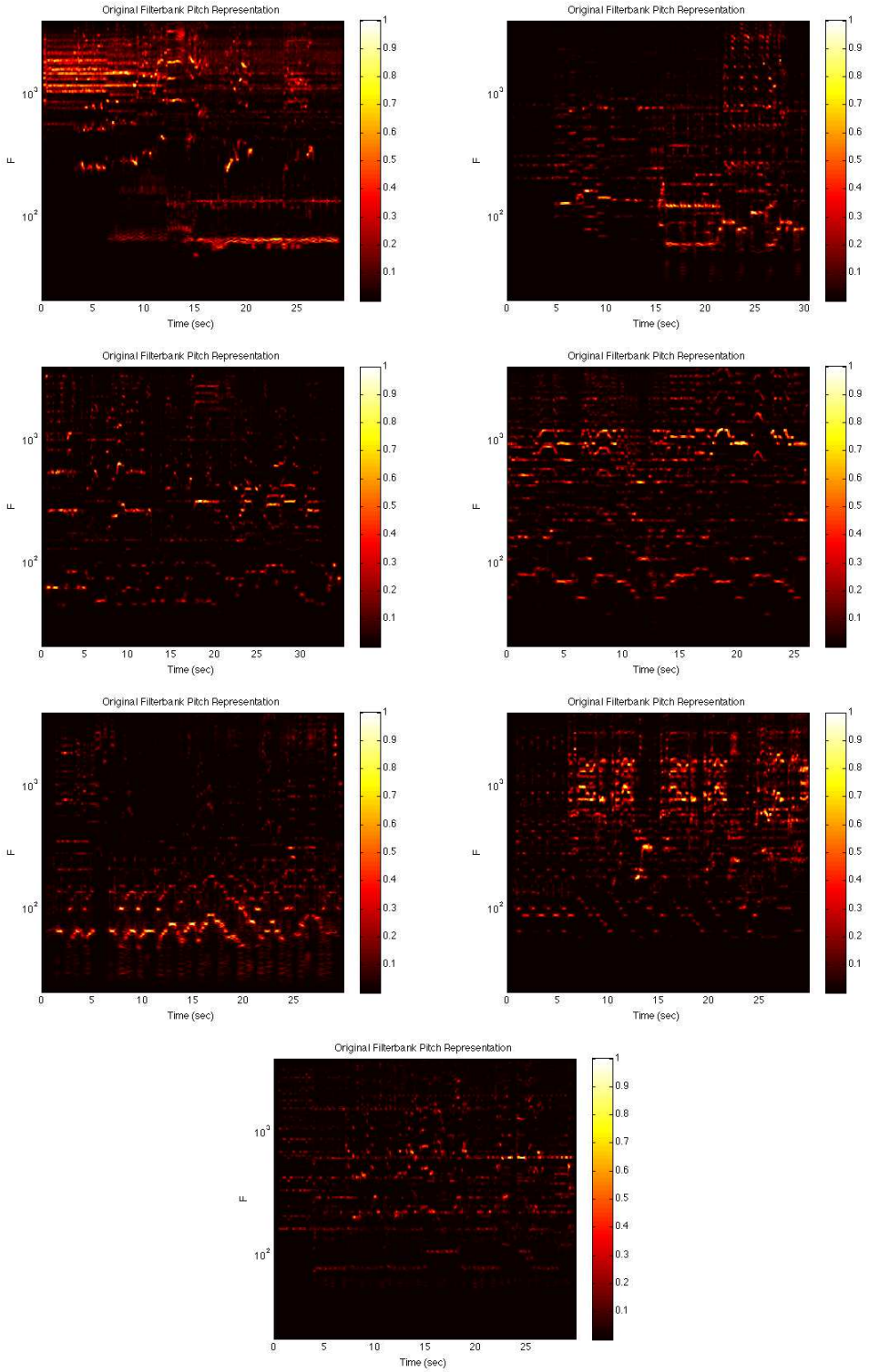

Figure A.2: Spectrograms of testfiles for chord recognition. **Left to right, top to bottom:** Testfiles 7 to 13, compare Table A.1.

Table A.2: Testfiles for onset detection.

Number	Title	Duration [s]	Remarks
1	Friedrich Burgmüller: Piano etude Op. 100, No. 2	17	l'Arabesque
2	Ludwig van Beethoven: 5th Symphony	21	Bernstein interpretation
3	Johannes Brahms: Hungarian Dance Op. 39, No. 5	15	
4	Happy Birthday	21	Choir accompanied by a piano
5	In Frankrijk buiten de poorten	8	Dutch folk song, female singing, see [11]
6	Ludwig van Beethoven: 5th Symphony	21	MIDI file
7	Alexander Borodin: String Quartet No. 2	38	Notturno
8	Dmitri Shostakovich: Waltz No. 2	30	Jazz Suite No. 2, Yablonsky interpretation
9	cello1	14	
10	clarinet1	30	
11	classic2	20	
12	classic3	14	
13	distguit1	6	
14	guitar2	15	
15	guitar3	15	
16	jazz2	14	
17	jazz3	11	
18	piano1	15	
19	pop1	15	
20	rock1	15	
21	sax1	12	
22	trumpet1	14	
23	violin2	15	

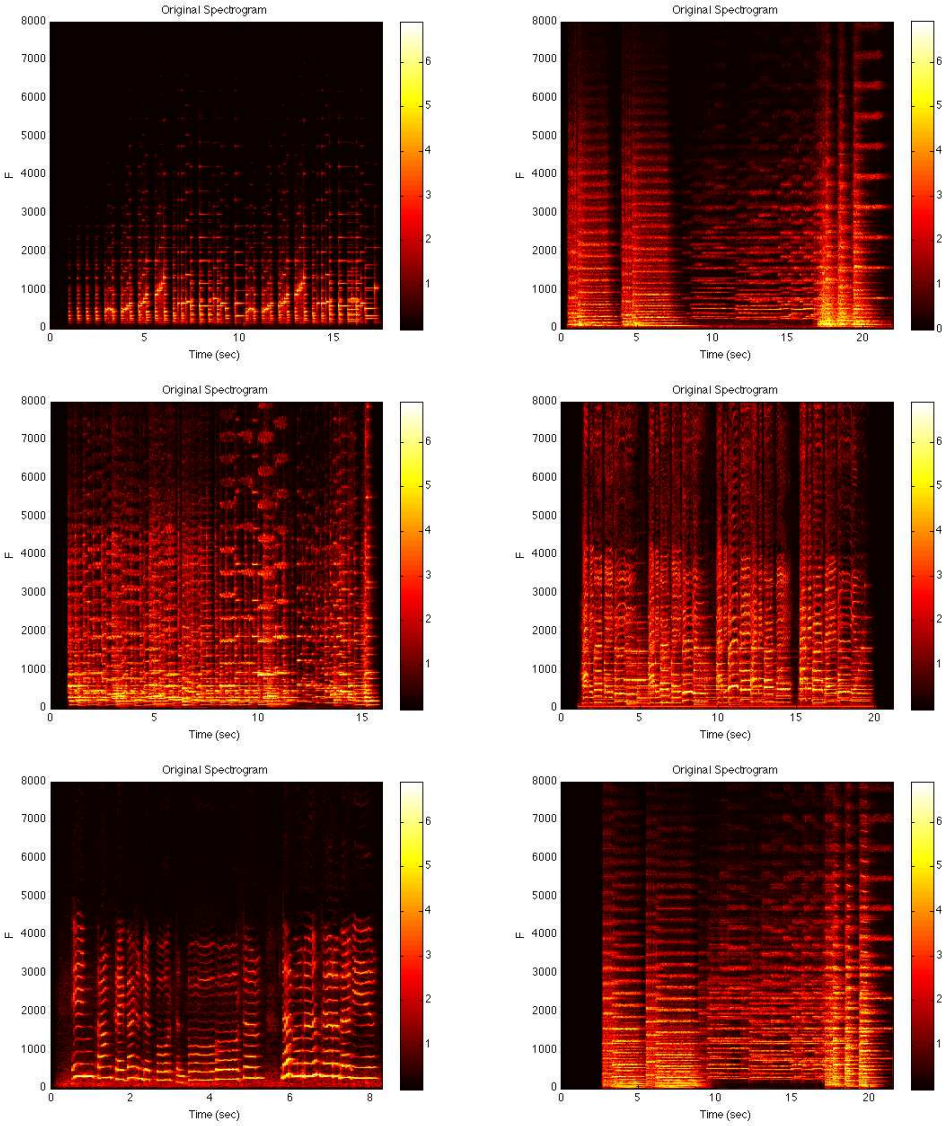

Figure A.3: Spectrograms of testfiles for onset detection. **Left to right, top to bottom:** Testfiles 1 to 6, compare Table A.2.

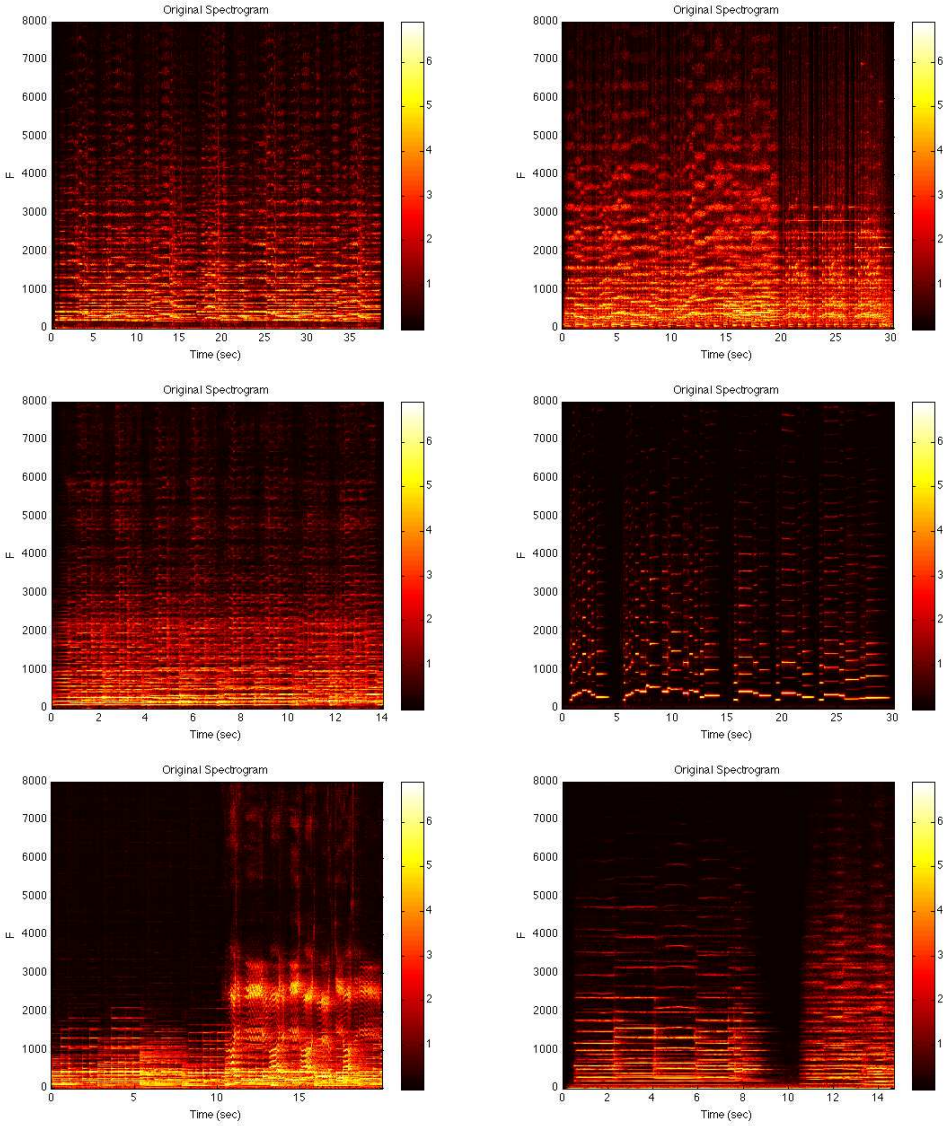

Figure A.4: Spectrograms of testfiles for onset detection. **Left to right, top to bottom:** Testfiles 7 to 12, compare Table A.2.

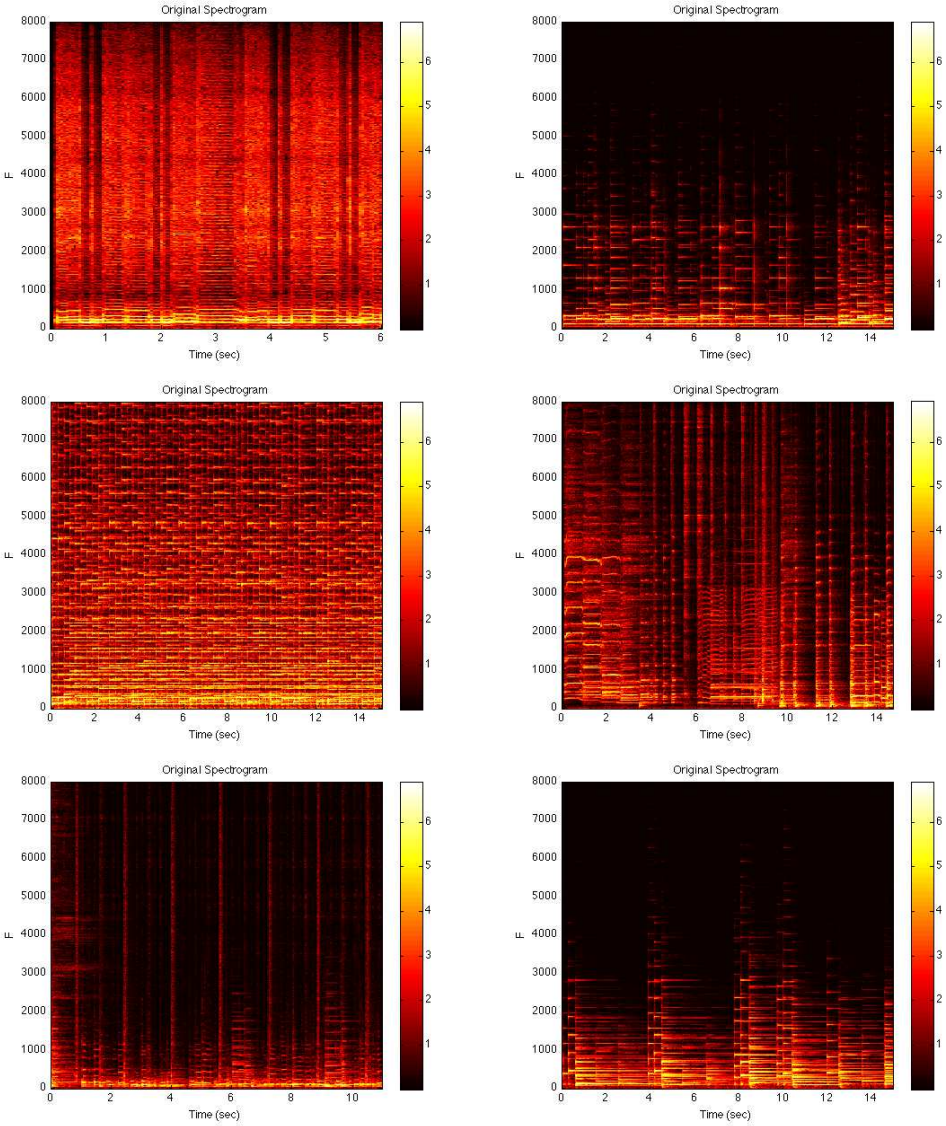

Figure A.5: Spectrograms of testfiles for onset detection. **Left to right, top to bottom:** Testfiles 13 to 18, compare Table A.2.

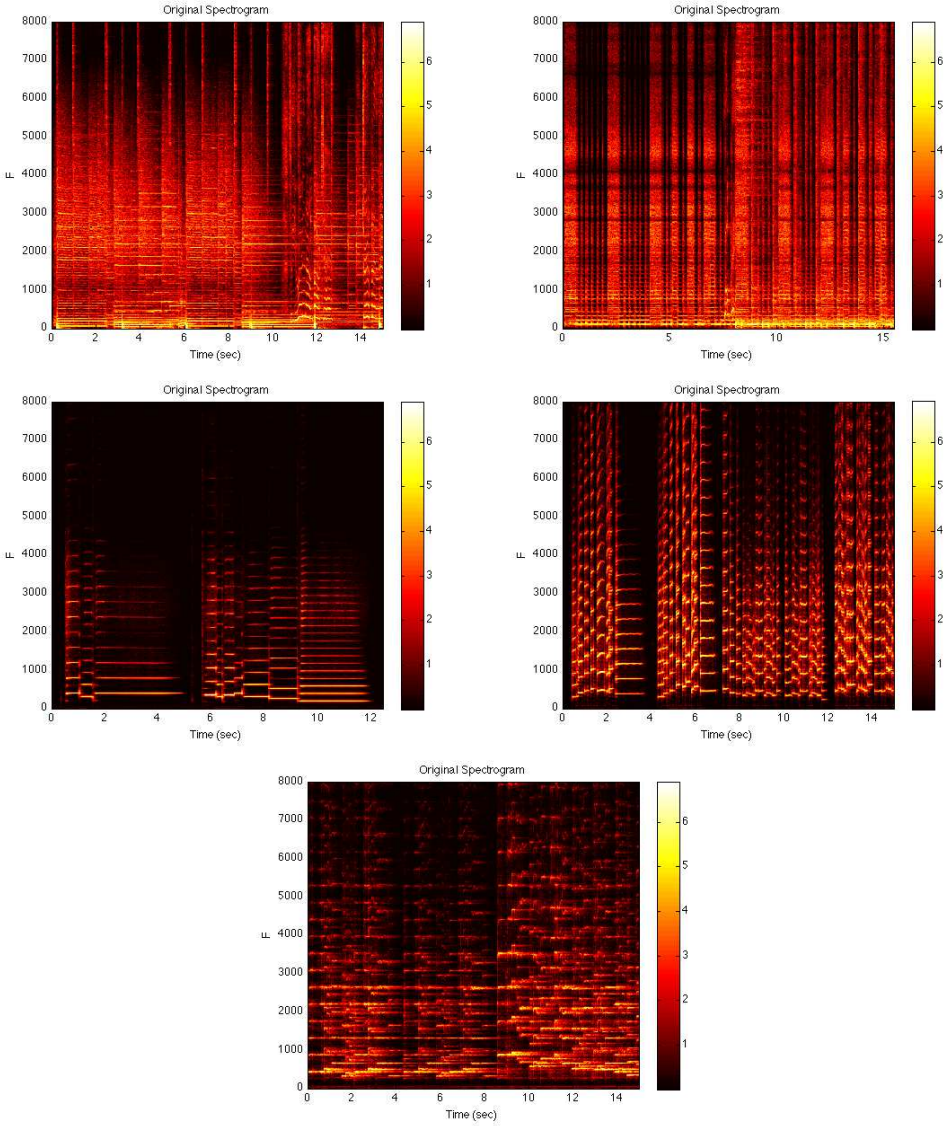

Figure A.6: Spectrograms of testfiles for onset detection. **Left to right, top to bottom:** Testfiles 19 to 23, compare Table A.2.

List of Figures

List of Tables

Bibliography

[1] Mathematical morphology (Wikipedia). http://en.wikipedia.org/wiki/Mathematical_morphology. Retrieved: 2010-04-07.

[2] MEX-files guide. http://www.mathworks.com/support/tech-notes/1600/1605.shtml. Retrieved: 2010-05-02.

[3] Precision, Recall and F-measure (Wikipedia). http://en.wikipedia.org/wiki/Precision_and_recall. Retrieved: 2010-04-02.

[4] RWC music database. http://staff.aist.go.jp/m.goto/RWC-MDB. Retrieved: 2010-06-10.

[5] Skeletonization examples. http://www.inf.u-szeged.hu/~palagyi/skel/skel.html. Retrieved: 2010-06-12.

[6] Ronald N. Bracewell. *The Fourier Transform and its Applications*. McGraw-Hill, New York, 3rd edition, June 1999.

[7] T. Brox and D. Cremers. Iterated nonlocal means for texture restoration. In F. Sgallari, A. Murli, and N. Paragios, editors, *Scale Space and Variational Methods in Computer Vision*, volume 4485 of *Lecture Notes in Computer Science*, pages 13–24, Berlin, 2007. Springer-Verlag.

[8] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 60–65. IEEE Computer Society, June 2005.

[9] L. Calabi and W. E. Hartnett. Shape recognition, prairie fires, convex deficiencies and skeletons. Parke Mathematical Laboratories, April 1968.

[10] H. T. Cheng, Y.-H. Yang, Y.-C. Lin, I-B. Liao, and H. H. Chen. Automatic chord recognition for music classification and retrieval. In *Proceedings of the 2008 IEEE International Conference on Multimedia and Expo, ICME 2008, June 23-26 2008, Hannover, Germany*, pages 1505–1508. IEEE, 2008.

[11] Dutch Song Database. In Frankrijk buiten de poorten. http://www.liederenbank.nl/liedpresentatie.php?zoek=73285&lan=en. Retrieved: 2010-06-14.

[12] S. Dixon. Onset detection revisited. In *Proc. of the 9th Int. Conference on Digital Audio Effects (DAFx-06)*, Montreal, September 2006.

[13] C. Duxbury, J. P. Bello, M. Davies, and M. Sandler. Complex domain onset detection for musical signals. In *Proc. of the 6th Int. Conference on Digital Audio Effects (DAFx-03)*, London, September 2003.

[14] S. Ewert, M. Müller, and P. Grosche. High resolution audio synchronization using chroma onset features. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, April 2009.

[15] M.A. Förstner and E. Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *Proc. ISPRS Intercommission Conf. on Fast Processing of Photogrammetric Data*, pages 281–305, Interlaken, June 1987.

[16] P. Grosche and M. Müller. Computing predominant local periodicity information in music recordings. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 33–36, New Paltz, New York, USA, 2009.

[17] P. Grosche, M. Müller, and S. Ewert. Combination of onset-features with applications to high-resolution music synchronization. In *Proceedings of the 35th International Conference on Acoustics (NAG/DAGA)*, pages 357–360, 2009.

[18] R. Kimmel, D. Shaked, N. Kiryati, and A. M. Bruckstein. Skeletonization via distance maps and level sets. *Computer Vision and Image Understanding*, 62(3):382–391, 1995.

[19] V. Konz, M. Müller, and S. Ewert. A multi-perspective evaluation framework for chord recognition. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, Utrecht, Netherlands, 2010, to appear.

[20] K. Lee and M. Slaney. Automatic chord recognition from audio using an HMM with supervised learning. In *Proc. 7th International Conference on Music Information Retrieval, ISMIR 2006*, 2006.

[21] W.-C. Lee and C. C. J. Kuo. Musical onset detection based on adaptive linear prediction. In *Proceedings of the 2006 IEEE International Conference on Multimedia and Expo, ICME 2006, July 9-12 2006, Toronto, Ontario, Canada*, pages 957–960. IEEE, 2006.

[22] P. Leveau, L. Daudet, and G. Richard. Methodology and tools for the evaluation of automatic onset detection algorithms in music. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Spain, 2004.

[23] M. Mahmoudi and G. Sapiro. Fast image and video denoising via nonlocal means of similar neighborhoods. *IEEE Signal Processing Letters*, 12:839–842, December 2005.

[24] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler. OMRAS2 metadata project 2009. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Kobe, Japan, 2009.

[25] M. Müller. *Information Retrieval for Music and Motion*. Springer-Verlag, Berlin, 1st edition, September 2007.

[26] J. Orchard, M. Ebrahimi, and A. Wong. Efficient nonlocal-means denoising using the SVD. In *Proceedings of the International Conference on Image Processing, ICIP*, pages 1732–1735. IEEE, 2008.

[27] N. Otsu. A threshold selection method from gray level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, 1979.

[28] P. Soille. *Morphological Image Analysis*. Springer-Verlag, Berlin, 2nd edition, 1999.

[29] K. Sumi, K. Itoyama, K. Yoshii, K. Komatani, T. Ogata, and H. G. Okuno. Automatic chord recognition based on probabilistic integration of chord transition and bass pitch estimation. In *Proc. 9th International Conference on Music Information Retrieval, ISMIR 2008*, pages 39–44, 2008.

[30] R. van den Boomgaard. The morphological equivalent of the Gauss convolution. *Nieuw Archief Voor Wiskunde*, 10(3):219–236, 1992.

[31] J. Weickert. Theoretical foundations of anisotropic diffusion in image processing. *Computing, Suppl. 11*, pages 221–236, 1996.

[32] J. Weickert. *Anisotropic Diffusion in Image Processing*. ECMI Series. Teubner-Verlag, Stuttgart, 1998. Out of print. Can be downloaded at `http://www.mia.uni-saarland.de/weickert/Papers/book.pdf`.

[33] J. Weickert. Coherence-enhancing diffusion filtering. *International Journal of Computer Vision*, 31:111–127, 1999.

[34] S. Zimmer, S. Didas, and J. Weickert. A rotationally invariant block matching strategy improving image denoising with non-local means. In *Proc. 2008 International Workshop on Local and Non-Local Approximation in Image Processing*, Lausanne, Switzerland, August 2008.

[35] J. J. Zou. A fast skeletonization method. In C. Sun, H. Talbot, S. Ourselin, and T. Adriaansen, editors, *Proc. 7th Digital Image Computing: Techniques and Applications*, pages 283–288, Sydney, December 2003.