Audiokodierung durch musikalische Fingerabdrücke

Diplomarbeit

Tobias Mur

Betreuer

Univ. Ass. DI Dr. Alois Sontacchi

Begutachter

o.Univ. Prof. Mag. DI Dr. Robert Höldrich

Institut für Elektronische Musik und Akustik
Universität für Musik und darstellende Kunst
Graz, Österreich
September 2007

Zusammenfassung

In dieser Arbeit wird eine Audiokodierung anhand so genannter "musikalischer Fingerabdrücke" untersucht. Solche Fingerabdrücke (thumbnails) sind für das jeweilige Musikstück repräsentative Teile (z.B. Refrain, Strophe...). Findet man Teile, die sich wiederholen, so muss man diese Teile nur mehr einmal speichern und kann sie einfach öfter verwenden. Man erhält also Bauteile des Stücks und einen "Bauplan" dazu. Der erste Schritt dazu ist das Finden einer Rasterung, anhand welcher man die Ähnlichkeiten innerhalb eines Stückes sucht. Hierzu kann entweder eine "Onset - Detection" - Detektierung von Punkten, an welchen Noten beginnen - oder eine Rhythmus - Synchronisierung herangezogen werden. Der nächste Schritt ist das Extrahieren von Eigenschaften um das Stück zu beschreiben. Dies sind im Falle dieser Diplomarbeit spektrale (MFCC's) und musikalische (Chroma; Konstant- Q) Koeffizienten. Sind diese Beschreibungsmerkmale gefunden, so werden basierend auf Ähnlichkeitsmaßen (z.B. Korrelation) Wiederholungen gesucht. Findet man Teile mit großer Übereinstimmung, so hat man einen Fingerabdruck gefunden, welcher als Baustein bei der Kodierung dient. Anhand einer Tabelle wird angegeben, an welcher Stelle der Bausstein eingesetzt werden muss. Zu jeder dieser Teilaufgaben werden mehrere Ansätze und Techniken untersucht und miteinander verglichen.

Abstract

This work aims to realize audio coding through "musical fingerprints". Such fingerprints (thumbnails) can be representative pieces of a song alike chorus, verse, etc. or even smaller ones. If we find repeated parts in a song, these parts have to be saved just once and can be used manifold. In this way one gets a set of components and a construction plan for the song. First of all we have to divide the song in a meaningful way to calculate the similarity between each defined part. Therefore onset detection, i.e. detection of the starting points for notes, or a beat tracking approach, i.e. rhythm synchronisation, can be used to find start and end-points of each part. Within the next step, for each of these parts features will be extracted that submit a musically description. In this thesis spectral (MFCC's) and musical (Chroma; constant - Q) coefficients are investigated. Found these characteristics, repetitions are searched due to similarity - measurement (e.g. correlation). If highly correlated parts are found, a thumbnail and hence a basic component for the coding is located and stored. Furthermore the temporal locations of the detected similar parts within the song are stored in a look-up table, too. For each of these tasks several approaches are tested and compared.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Die vorliegende Arbeit beschäftigt sich mit dem Thema der Audiokodierung durch sogenannte musikalische Fingerabdrücke. Dieser Ansatz basiert darauf, dass sich wiederholende Teile eines Musikstückes bei der Kodierung nur einmal berücksichtigt (gespeichert) werden müssen, um dann bei der Dekodierung an den entsprechenden Stellen wieder eingesetzt zu werden. Wiederkehrende Teile in einem Musikstück können als Fingerabdrücke (*Fingerprints, Thumbnails*) bezeichnet werden, da sie einen charakteristischen Abschnitt darstellen.

1.1 Einführung in die Thematik

Um *Thumbnails* zu finden werden verschiedene Schritte durchlaufen (siehe Abbildung 1.1[1]) . Da der Beginn eines Abschnittes in der Musik zwangsläufig mit dem Einsetzen einer Note zusammenhängen muss, wird als erstes versucht, Startpunkte der verschiedenen Noten zu finden. Hierzu wird eine *onset-detection* durchgeführt, anhand derer diese Punkte gefunden werden. Im Vergleich dazu wird auch eine *beat-detection* (Rhythmuserkennung) durchgeführt, da davon ausgegangen werden kann, dass der Beginn eines Fingerabdruckes mit dem rhythmischen Grundmuster zusammenhängt. Durch diese beiden Verfahren kann eine Segmentierung (*framing*) vor-

[1]Der im Blockschaltbild strichliert eingezeichnete Weg war zwar anfangs geplant, wurde dann aber nichtmehr getestet.

genommen werden. Nun werden jeweils zwischen zwei Noten-Startpunkten, beziehungsweise zwei Taktschlägen, verschiedene Beschreibungsmerkmale extrahiert. Es können dies musikalische (Tonhöhe, Ton) oder spektrale Eigenschaften (Frequenz) sein. Mit Hilfe dieser Merkmale wird das Stück beschrieben, da man zu jedem Zeitpunkt Angaben über den musikalischen Inhalt hat. Diese beschreibenden Eigenschaften werden nun innerhalb des Stückes verglichen um Sequenzen zu finden, die sich ähneln. Zu diesem Zweck kann zum Beispiel eine Korrelation durchgeführt werden. Anhand des Ähnlichkeitsmaßes kann man nun sich ähnelnde Teile finden und als *Thumbnails* ausweisen. Sind sich zwei Teile zu 100% ähnlich, d.h. man hat zwei identische Teile - eine Wiederholung - gefunden, so kann man diese in genannter Kodierung nur einmal Abspeichern und als Dekodierungsvorschrift einen Bauplan verwenden in dem die Zeitpunkte, an denen der Fingerabdruck zu verwenden ist, aufgelistet sind.

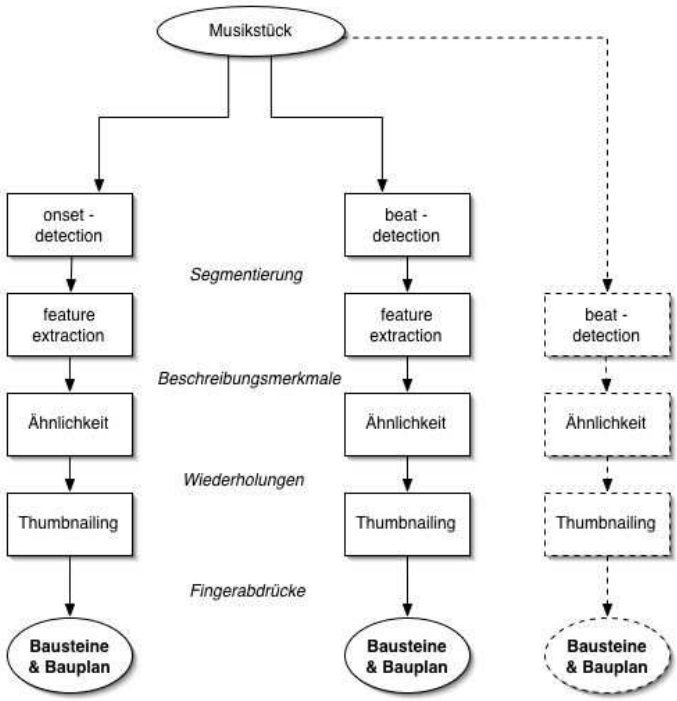

Abbildung 1.1: Blockschaltbild der Kodierung

Obige Abbildung (1.1) zeigt die verschiedene Pfade des Systems an. Dabei unterscheiden sich die zwei linken Pfade dadurch, dass für den Einen die *onset-detection*,

sprich die Noten-Startpunkte für das *framing* herangezogen werden und beim Zweiten der *beat*, sprich die Taktschläge. Der dritte, strichlierte Pfad steht für eine Suche nach sich wiederholenden Rhythmusmustern, welche jedoch nicht als eigener Ansatzt zum Finden von Wiederholungen herangezogen wurde.

1.2 Ziel der Arbeit

Ziel dieser Arbeit ist es, für das oben genannte Verfahren den bestmöglichen Weg zu finden. Dazu werden für jeden Schritt mehrere Ansätze aus der vorhandenen Literatur angeführt und miteinander verglichen. Eine Evaluierung der verschiedenen Methoden soll zeigen, welcher Ansatz am besten funktioniert, bzw. wie groß die Abhängigkeit von der Art der Musik ist. Im Rahmen der Evaluierung werden Musikstücke aus verschiedenen Genres mit verschiedenen Eigenschaften als Testobjekte verwendet. Das Hauptaugenmerk liegt dabei auf der Popularmusik (Pop), da für dieses Genre die untersuchte Kodierung voraussichtlich das größte Potential liefert.

1.3 Struktur

Dem Blockschaltbild aus Abbildung (1.1) folgend, wird zuerst die *onset-detection* behandelt. Hierzu werden mehrere Ansätze zur Detektion von *onsets* vorgestellt und miteinander verglichen. Zuerst wird hier auch ein *pre-processing* durchgeführt beziehungsweise erklärt. Als *post-processing* wird ein *peak-picking* in den erzielten *onset*-Funktionen beschrieben. Als nächstes Kapitel folgt die *feature-extraction*, in der Ansätze zum Extrahieren der Rhythmustruktur, spektraler und musikalischer Merkmale beschrieben werden. In Kapitel 4 wird die Berechnung der Ähnlichkeitsmatrix und die Bestimmung von wiederholten Teilen vorgestellt. Kapitel 5 behandelt die einzelnen Tests und die dazugehörigen Berechnungen und Auswertungen werden hier erklärt. Die Auswertung der Ergebnisse und der Vergleich der verschiedenen Ansätze und der erzielten Ergebnisse erfolgt im 6. Kapitel. Im abschließenden Kapitel

wird die Arbeit zusammengefasst, sowie ein Ausblick auf zukünftige Verbesserungsmöglichkeiten gegeben.

Kapitel 2

Onset detection

Bei der *onset-detection* wird das Ziel verfolgt, jene Punkte im Musiksignal zu finden, an denen Noten beginnen [12]. Ausgehend von diesen *onsets* kann man auf das rhythmische Grundmuster schließen. Im Falle dieser Arbeit werden die *onsets* jedoch als Grundraster für die Berechnung der beschreibenden Merkmale verwendet.

Ausgehend vom originalen Musiksignal wird vorerst ein sogenanntes *pre-processing*, also eine Signalaufbereitung vorgenommen, um gewisse Eigenschaften (z.B. transiente Übergänge, hochfrequente Anschlaggeräusche, usw.) zu akzentuieren. Nachdem eine *onset*-Funktion berechnet ist, wird in dieser ein *peak-picking* vorgenommen, um die tatsächlichen Startpunkte der Noten zu finden. Nachfolgende Abbildung (2) zeigt die Schritte zur *onset-detection* vom Signal zu den gesuchten *onsets*.

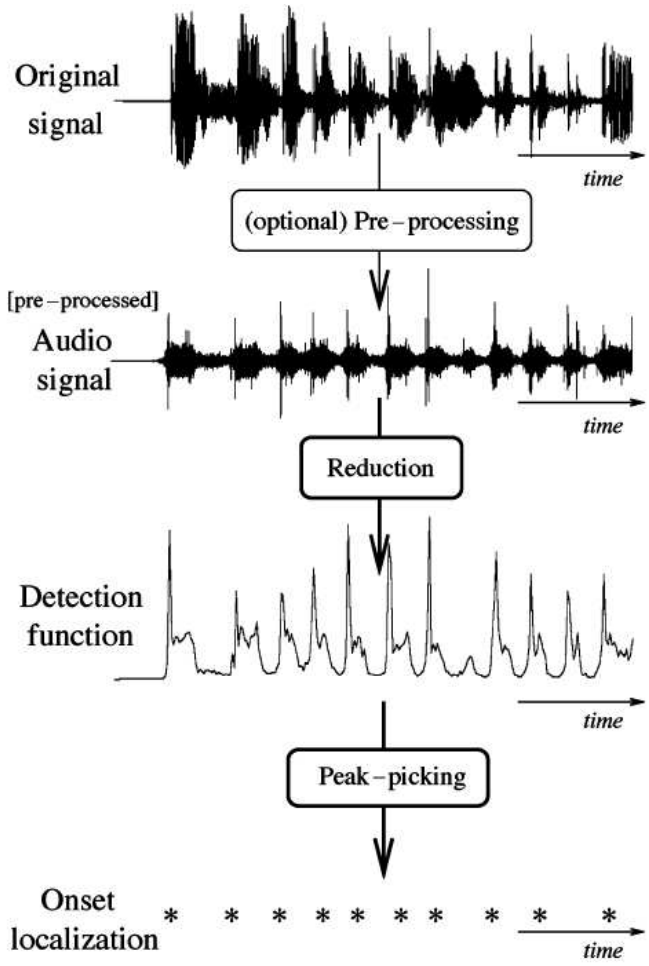

Abbildung 2.1: Flussdiagramm zur *onset-detection* [12]

Betrachtet man eine Note als transientes Ereignis (Abbildung 2), so kann man sie in drei Teile unterteilen: *Onset, Attack* und *Decay. Onset* ist der Punkt, an dem der Anstieg der Amplitude beginnt, *attack* ist jener Anstieg und *decay* ist der Bereich, während dessen die Einhüllende der Amplitude wieder abfällt:

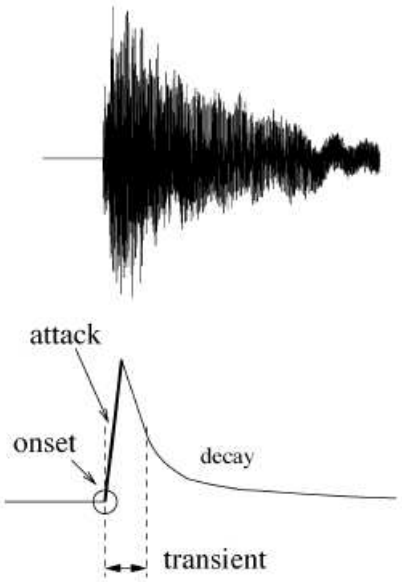

Abbildung 2.2: *Onset*, *attack* und *decay* am Beispiel einer einzelnen Note [12]

2.1 Pre - Processing

Beim *pre-processing* wird das Signal so transformiert, dass für das Auffinden von *onsets* relevante Eigenschaften hervorgehoben werden. Dies können mehrere Eigenschaften sein. Am besten dafür geeignet sind die Trennung von stabilen und transienten Zuständen und das Aufteilen des Signales in mehrere Frequenzbänder (vgl. dazu [8, 11, 12]. Das *pre-processing* ist nicht zwingend notwendig, erhöht aber meist die Zuverlässigkeit des Auffindens der *onsets* und somit die Funktionalität der *onset-detection*.

2.2 basierend auf zeitlichen Eigenschaften

Beobachtet man ein Musiksignal, so ist in den meisten Fällen klar erkennbar, dass der Beginn einer Note eine Erhöhung der Signalamplitude mit sich führt. Zieht man dies in Betracht, so kann man *onset-detection* erziehlen, in dem man einfach die zeitliche Einhüllende des Signals betrachtet. Dies kann zum Beispiel durch Be-

trachtung des Betrags und anschließende Glättung erzielt werden [12], was eine vereinfachte bzw. approximierte Form der Einhüllenden darstellt[1]:

$$E_0(n) = \frac{1}{N} \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} \mid x(n+m) \mid w(m). \tag{2.1}$$

Hierbei ist $w(m)$ ein Fenster der Länge N, welches zur Glättung dient. Ein solcher *envelope follower* erzielt gute Ergebnisse für stark perkussive Signale. Bei Signalen mit nicht perkussiv ausgeprägten Eigenschaften ist dieser Ansatz jedoch nicht gut geeignet. Eine Variation dieser Funktion ist eine Betrachtung der Energie, anstelle der Amplitude. Die mittlere lokale Energie kann wie folgt bestimmt werden(vgl. [12]):

$$E(n) = \frac{1}{N} \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} [x(n+m)]^2 \, w(n). \tag{2.2}$$

Bei Versuchen hat sich jedoch gezeigt, dass das Quadrieren des Signals eher eine Verschlechterung für die *onset-detection* darstellt (vgl. [12]).

Eine weitere Verbesserung stellt die Verwendung eines leckenden Tiefpasses bzw. Integrators dar:

$$y(n) = \alpha y(n-1) + (1-\alpha)x(n) \tag{2.3}$$

Verwendet man einen schnellen ($\alpha_1 = 0.9$) und einen langsamen ($\alpha_2 = 0.999$) Integrator und bildet die Differenz der beiden Signale, erhält man eine Signalfolge, welche an den *onsets* deutliche Spitzen aufweist.

[1] Die exakte Berechnung der Einhüllenden kann durch Bildung des analytischen Signals (beispielsweise mittels des Hilbert-Transformers) erfolgen.

2.3 basierend auf spektralen Eigenschaften

Betrachtet man den spektralen Verlauf von Musiksignalen, so erkennt man, dass an den Startpunkten von Noten (*onsets*) hochfrequente Komponenten auftreten. Dies geschieht meistens infolge von Anschlag- und Anzupfgeräuschen, aber auch bei Anblasgeräuschen. Nutzt man diese Eigenschaften aus, so erhält man relativ gute Ergebnisse zur *onset-detection* für eine relative breit gefächerte Anzahl von Signalen. Für die auf spektralen Eigenschaften basierenden Ansätze muss das jeweilige Signal zuerst in den Frequenzbereich transformiert werden. Hierfür wird die Fourrier - Transformation verwendet:

$$X_k(n) = \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} x(nh+m)w(m)e^{-j\frac{2\pi km}{N}} \tag{2.4}$$

wobei $w(m)$ wieder die Fensterfunktion und h die *hopsize* ist ; $k = 0$ bis $N-1$ repräsentiert die Bin-Nummer.

Da Transiente im Zeitbereich durch relativ kurzzeitige, starke Amplitudenänderungen charakterisiert werden, ergibt sich im Spektralbereich ein kurzzeitiger, breitbandiger Anstieg im Amplitudengang. Dies kann nun für die *onset-detection* verwendet werden, indem man sich die hochfrequenten Anteile ansieht (HFC):

$$\tilde{E}(n) = \frac{1}{N}\sum_{k=0}^{N-1} W_k \mid X_k(n) \mid^2 . \tag{2.5}$$

Dabei stell W_k eine Gewichtungsfuntkion für die Frequenzbins dar. Ist $W_k = 1 \forall k$, so erhält man die lokale Energie. Mit $W_k = k^2 \, \forall k$ ergäbe sich die lokale Energie des abgeleiteten Signals. Die eigentliche *high frequency content* - Funktion ergibt sich laut Definition für $W_k = k \, \forall k$, welche die höheren Frequenzen stärker gewichtet als tiefere [12].

Eine weitere Möglichkeit *onsets* anhand von spektralen Betrachtungen zu detektieren ist die *spectral difference* oder *spectral flux.* Hierbei wird die spektrale Differenz zwischen zwei aufeinanderfolgenden Spektren berechnet:

$$SD(n) = \sum_{k=0}^{N-1} H\left(\mid X_k(n) \mid - \mid X_k(n-1) \mid\right) \tag{2.6}$$

mit $H(x) = \frac{x+|x|}{2}$, was eine Halbwellengleichrichtung der Funktion ergibt. Je größer diese Differenz der Spektren ist, desto größer ist die Wahrscheinlichkeit, dass ein onset stattgefunden hat, weil sich offensichtlich das Spektrum des Signals geändert hat. Anzumerken ist, dass bei verschiedenen Tests die L_1 - bzw. L_2 - Norm herangezogen wurden, wobei sich die L_1- Norm als bessere Wahl herausgestellt hat [10]. Ebenfalls kann man auch hier die spektralen Differenzen zusätzlich frequenzabhängig gewichten.

2.4 basierend auf der Phase

Einen weiteren Indikator für einen abrupten Wechsel im Signal stellt die Phase dar, da der Beginn einer Note meistens einen Sprung in der Phase mit sich bringt. Dazu betrachtet man die Fourriertransformierte als

$$X_k(n) = \mid X_k(n) \mid e^{j\varphi_k(n)} \tag{2.7}$$

wobei $-\pi < \varphi_k(n) \leq \pi$. Geht man davon aus, dass sich die Phase im Normalfall nur gering beziehungsweise sehr langsam ändert, so kann man folgende Gleichung ausfstellen:

$$\varphi_k(n) - \varphi_k(n-1) \simeq \varphi_k(n-1) - \varphi_k(n-2). \tag{2.8}$$

Die Abweichung der Phase kann als Differenz der Differenzen angeschrieben werden (als Differenzengleichung 2. Ordnung):

$$\Delta\varphi_k(n) = \varphi_k(n) - 2\varphi_k(n-1) + \varphi_k(n-2) \simeq 0. \tag{2.9}$$

Diese Abweichung wird während transienten Vorgängen relativ große Werte annehmen, da hier die Phase nicht gut definiert ist und meisten ein Sprung in der Phase

auftritt. Summiert man über alle Frequenzen diese Abweichung auf und normiert auf die Anzahl der Frequenzen, erhält man einen guten Wert für die Änderung der Phase und damit der Frequenz zu jedem Zeitpunkt (Phase Deviation [10, 12]):

$$PD(n) = \frac{1}{N} \sum_{k=0}^{N-1} \mid \Delta\varphi_k(n) \mid . \tag{2.10}$$

Um Einflüsse von Rauschen zu unterdrücken, das normalerweise in der Amplitude kleiner ist als das Signal selbst, werden die eizelnen Frequenzbins, beziehungsweise die dazugehörigenden Phasendifferenzen mit der Amplitude des Frequenzbins gewichtet. Die resultierende Funktion nennt sich *Weighted Phase Deviation WPD* [10]:

$$WPD(n) = \frac{1}{N} \sum_{k=0}^{N-1} \mid X_k(n)\Delta\varphi_k(n) \mid. \tag{2.11}$$

Dixon [10] stellt eine weitere Möglichkeit vor, um die störenden Einflüsse zu verringern. Dies ist die auf die Frequenzbins normalisierte *WPD* (*NWPD*):

$$NWPD(n) = \frac{\sum_{k=0}^{N-1} \mid X_k(n)\Delta\varphi_k(n) \mid}{\sum_{k=0}^{N-1} \mid X_k(n) \mid}. \tag{2.12}$$

2.5 im spektralen Bereich

Dixon zeigt in [10] *onset detection* im spektralen Bereich, wobei ein erwarteter Wert für das Signal ($\hat{X}_k(n)$) aus den vergangenen Werten ($X_k(n-1)$; $X_k(n-2)$) berechnet wird und vom wirklichen Wert ($X(n,k)$) abgezogen. Stimmen der geschätzte und der tatsächliche Wert nicht überein, wird ein Onset detektiert. Der erwartete Wert wird dabei wie folgt berechnet:

$$\hat{X}_k(n,k) = \mid X_k(n-1) \mid e^{j[\varphi_k(n-1)+\Delta\varphi_k(n-1)]} \tag{2.13}$$

mit $\Delta\varphi(n) = \varphi(n) - \varphi(n-1)$.

Aus der Differenz zwischen geschätztem und tatsächlichem Wert wird nun die Funktion zur Onset-detection berechnet:

$$CD(n) = \sum_{k=0}^{N-1} \mid X_k(n) - \hat{X}_k(n) \mid . \tag{2.14}$$

Bello [12] zeigt eine ähnliche Berechnung in der Spektrum, jedoch wird hier die Differenz als euklidische Distanz zwischen wahrem und berechnetem Wert angeschrieben:

$$\Gamma_k(n) = \sqrt{\mid \hat{X}_k(n) \mid^2 + \mid X_k(n) \mid^2 -2 \mid \hat{X}_k(n) \mid\mid X_k(n) \mid \cos(\Delta\varphi_k(n))} \tag{2.15}$$

mit $\Delta\varphi_k(n) = \varphi_k(n) - 2\varphi_k(n-1) + \varphi_k(n-2)$

Die *onset-detection*-Funktion erhält man dann durch aufsummieren der Distanzen über die Frequenzen:

$$\zeta(n) = \sum_{k=1}^{N} \Gamma_k(n) \tag{2.16}$$

In dieser Arbeit kommt dieser Ansatz in Zusammenhang mit hybriden Ansätzen (Kapitel 2.7) vor [8,11].

2.6 Heuritische Verfahren (Informationstheoretisch)

Der Ansatz von Abdallah [1] geht davon aus, dass *onsets* überraschende Ereignisse sind. Beginnt eine Note, so wird sich das Wahrscheinlichkeitsmodell des Signales ändern. Verfolgt man dieses Modell über die Zeit, so wird man jeweils an denjenigen Punkten, an denen *onsets* stattgefunden haben, Peaks ergeben. Die Überraschung wird dabei als der negative Logarithmus der Wahrscheinlichkeitsdichte (log-Likelihood) P definiert:

$$S(x) = -\log P(x). \tag{2.17}$$

Ausgehend von einer Gauß'schen Wahrscheinlichkeitsdichtefunktion

$$P(x) = \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x_i^2}{2\sigma^2}} \tag{2.18}$$

(Mittelwertfreie Realisierung mit der Varianz σ) ergibt sich für die Überraschung folgende Gleichung:

$$S(x) = \frac{1}{2\sigma^2} \sum_{i=1}^{n} x_i^2 + const. \tag{2.19}$$

Qualitativ ergibt dies dieselbe Detektionsfunktion wie in Gleichung 2.2. Möchte man nun die Abhängigkeit der einzelnen benachbarten Samples mittels eines Autoregressiven Modells (AR) beschreiben, kann man die Kovarianz Matrix $C = E[xx^T]$ verwenden. Dadurch lässt sich die Wahrscheinlichkeitsfunktion aus Gleichung 2.18 wie folgt anschreiben:

$$P(x) = \frac{e^{-\frac{1}{2}x^T C^{-1} x}}{\sqrt{(2\pi)^n detC}}. \tag{2.20}$$

Dadurch kann die Überraschung durch Matrizen-Multiplikation für jeden Signalausschnitt berechnet werden. Die Konstante (*const.*) kann dabei vernachlässigt werden.

$$S(x) = -\frac{1}{2} x^T C^{-1} x + const. \tag{2.21}$$

Eine Möglichkeit, eine kompakte Notation durch die Eigenvektoren zu erhalten ist die Singulärwertzerlegung. Obige Formel ergibt sich somit zu:

$$S(x) = \sum_{i=1}^{n} \frac{\left(u_i^T x\right)^2}{2\sigma_i^2} + const. \tag{2.22}$$

Nun wird das Musiksignal anstatt in sequentiell aufeinanderfolgende Signalauschnitte in sich überlappende Teile unterteilt ($x[1]$, $x[2], \ldots, x[N]$), wobei sich ein Ausschnitt $x[i]$ nochmals in $x_1[i]$ und $x_2[i]$ aufteilt, wie in Abbildung 2.3 dargestellt.

Aus der Verbundwahrscheinlichkeit $P(x_1, x_2) = P(x)$ kann nun die bedingte

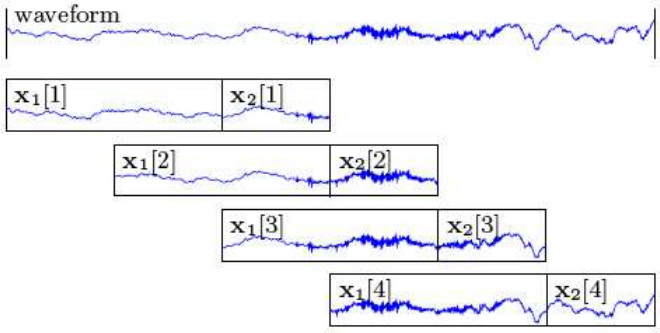

Abbildung 2.3: Aufteilung des Signals in überlappenden Abschnitte

Wahrscheinlichkeit

$$P(x_2 \mid x_1) = \frac{P(x_1, x_2)}{P(x_1)} = \frac{P(x)}{P(x_1)} \tag{2.23}$$

berechnet werden. Aus dieser bedingten Wahrscheinlichkeit wird wieder der negative Logarithmus gebildet, um zum jeweiligen Wert der „Überraschung" zu gelangen:

$$S(x) = \log P(x_1) - \log P(x) = \sum_{i=1}^{n} \frac{\left(u_{1_i}^T x_1\right)^2}{2\sigma_{1_i}^2} - \sum_{i=1}^{n} \frac{\left(u_i^T x\right)^2}{2\sigma_i^2}. \tag{2.24}$$

D.h., man bildet die Differenz aus negativem log-Likelihood des gesamten Frames und jenem des ersten Teils des selben Frames. Ergibt sich eine kleine Differenz, so hat sich das Signal nicht verändert, es hat keine Überraschung stattgefunden. Ist dieser Unterschied jedoch groß, so hat sich das in seiner Verteilung geändert und man hat eine Überraschung (einen *onset*) gefunden. Bei der Singulärwertzerlegung kann dabei nach jenem i-ten Wert abgebrochen werden, ab dem die Varianz zu einem gewünschten Prozentsatz durch die Eigenwerte erklärt ist.

2.7 Hybride Ansätze

Unter hybriden Ansätzen versteht man Mischformen, bei denen mehrere Ansätze in Kombination vorkommen (vgl. [11]). Im Fall der *onset-detection* wird das Eingangssignal in mehrere Frequenzbänder unterteilt, die jeweils eine verschiedene zeitliche

Auflösung haben (*multiresolution analysis*). Das Signal wird dabei mittels kaskadierten QMF - (*qudrature mirror filter*) in Subbänder unterteilt. Hierdurch kann man für die verschiedenen Frequenzbereiche individuell die zeitliche Auflösung verändern. Zum Beispiel ist es wünschenswert, bei den tiefen Frequenzen eine gute Frequenzauflösung durch größere Fensterlängen zu haben. Andererseits ist für die hohen Frequenzanteile eine kurze FFT-Fensterlänge und dadurch eine gute zeitliche Auflösung erstrebenswert, um schnelle, transiente Änderungen im Signal möglichst genau detektieren zu können. Somit kann in jedem Frequenzband ein anderer Ansatz zur *onset-detection* angewandt werden.

In [8] wird ein QM-Filter verwendet, um das Signal in die Subbänder aufzuteilen. QMF's sind Filterpaare G_0 und G_1 mit den Eigenschaften:

$$G_1(z) = G_0(-z) \tag{2.25}$$

$$G_1(\omega) = G_0(\omega - \pi). \tag{2.26}$$

Das bedeutet G_0ist eine modulierte Version von G_1und deshalb kann obige Gleichung wie folgt umgeschrieben werden:

$$G_1(0.5\pi + \omega) = G_0(0.5\pi - \omega). \tag{2.27}$$

Kaskadiert man mehrere solcher Filterpaare und halbiert die Abtastsrate nach jedem Filterausgang, sprich vor dem nächsten Filtereingang, erhält man eine sogenannte *constant-Q* Filterbank.

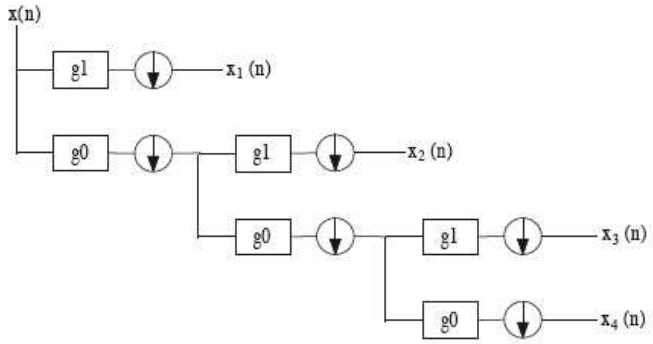

Abbildung 2.4: Kaskadiertes konstant - Q Filter. $x_{1...4}(n)$ sind die einzelnen Subbänder.

Nun hat man mehrere Subbänder mit verschiedener Auflösung und kann auf jedes der Bänder eine *onset-detection* anwenden und die erhaltenen Funktionen aufsummieren. Man könnte auch ein heuristisches Modell verwenden, sodass z.B. ein *onset* markiert wird, wenn in der Mehrzahl der Bänder ein Peak steht.

In [8] wird in allen Subbändern die Funktion aus Kapitel 2.5 angewandt. Eine Verbesserung stellt eine Verwendung mehrerer verschiedener Verfahren zur Detektion von *onsets* dar. Zum Beispiel kann für die hohen Frequenzbänder die spektrale Differenz (siehe dazu Kapitel 2.3) verwendet werden, während für tiefe Frequenzen die Berechnung der euklidischen Distanz aus Kapitel 2.5 verwendet wird [11].

2.8 Linear Prediction Coding

Um abrupte Änderungen in einem Signal zu finden, kann auch *linear-prediction-coding* (LPC) verwendet werden [13]. Diese basiert auf der Vorhersage des jeweiligen Samples aus vorhergegangenen Samples:

$$\tilde{x}(n) = \sum_{i=1}^{M} a_i x(n-i) \tag{2.28}$$

Bildet man nun die Differenz aus tatsächlichem Wert und geschätztem, so ergibt

sich der Vorhersagefehler zu

$$\varepsilon(n) = x(n) - \tilde{x}(n) = x(n) - \sum_{i=1}^{M} a_i x(n-i) \tag{2.29}$$

Aus diesem Fehler kann man nun bereits die *onset-function* bilden: dort wo der Fehler klein ist, ist das Signal annähernd gleich geblieben, dort wo sich ein großer Vorhersagefehler ergibt, hat sich das Signal zu stark geändert, um von der LPC richtig vorhergesagt werden zu können. Dieses Fehlersignal kann man nach kurzzeitigen Energieanstiegen durchsuchen und daraus das *onset*-Signal ableiten.

2.9 Postprocessing

Um aus den erlangten Funktionen *onsets* zu detektieren, muss ein Schwellwert (*threshold*) festgesetzt werden, anhand welchem festgelegt wird, welche Peaks der Funktion onsets sind und welche nicht. Hierfür gibt es zwei Möglichkeiten[2]:

1. man verwendet einen fixen Schwellwert
2. der Schwellwert ist adaptiv

Bei Verwendung eines fixen Schwellwerts werden onsets definiert, sobald ein festgelegter Wert überschritten wird: $d(n) \geq \delta$. $d(n)$ ist dabei die detection function und δ ein festegelegter, fixer Schwellwert. Der Nachteil hierbei ist jedoch, dass Änderungen in der Dynamik des Musiksignales nicht berücksichtigt werden. Aus diesem Grund ist es erforderlich, den Schwellwert an das Signal bzw. an die *detection-function* anzupassen. Zu diesem Zweck wird die Funktion meistens mittels eines Filters geglättet und diese geglättete Version als adaptiver threshold verwendet. Zum Beispiel kann ein Tiefpass-Filter zur Glättung eingesetzt werden:

$$\tilde{\delta}(n) = \delta + \sum_{i=0}^{M} a_i d(n-i). \tag{2.30}$$

[2]Des Weiteren kann man z.B. bei den Energiemethoden, wo die Onsetfunktion relativ glatt verläuft auch eine weitere Differentation durchführen

Alternativ dazu kann auch eine nichtlineare Glättung verwendet werden, zum Beispiel indem man die *onset*-Funktion quadriert und filtert. Solche Schwellwerte können jedoch großen Schwankungen unterliegen, sobald große peaks vorhanden sind und es können sogar kleinere peaks verdeckt werden. Eine bessere Methode zur Berechnung eines signalangepassten, adaptiven Schwellwertes stell die Lösung durch ein Medianfilter[3] dar:

$$\tilde{\delta}(n) = \delta + \lambda \, median \left\{ \mid d(n-M) \mid, ..., \mid d(n+M) \mid \right\}. \tag{2.31}$$

δ und λ sind dabei experimentell von Hand anzupassen und einzustellen [12]. Das Filter muss nicht zwangsläufig akausal in die Zukunft vorausblicken, doch in [12] findet sich das Filter nach Gleichung (2.31).

Den Nachteil des Medianfilters stellt der exzessive Berechnungsaufwand dar. Hierbei handelt sich um einen nichtlinearen Prozess, der zu einem Zeitpunkt n eine Anzahl von $2M + 1$ benachbarten Datenpunkten der Größe nach ordnet und jenen an der Stelle $M + 1$ befindlichen Samplewert ausgibt.

Zieht man den so berechneten Wert (den Medianwert) von der *onset*-Funktion ab, so erhält man ein Signal, aus dem es einfacher ist, die Spitzen zu finden. Es bleibt eine Funktion stehen, die nur an den Stellen, wo *onsets* stattfinden, größer als Null ist. Durch Halbwellengleichrichtung werden die Werte kleiner Null entfernt. Mit Hilfe der Kurvendiskussion sucht man die Spitzen in diesem Signal und markiert die Positionen dieser Spitzen als *onset*-Positionen. Um zu verhindern, dass *onsets* zu nahe hintereinander folgen, beziehungsweise dass auch irrelevante Peaks als *onsets* detektiert werden, werden die gefundenen Peaks mit dem jeweiligen Wert der *onset*-Funktion gewichtet, und innerhalb des gewünschten Zeitraumes (zum Beispiel 72ms, also Sechzehntelnoten bei ca. 200bpm) nur der jeweils dominanteste Peak als *onset* gewertet.

[3] Das Medianfilter wird vorallem in der Bildverarbeitung verwendet.

Kapitel 3

Feature Extraction

In der Feature Extraction wird versucht, Beschreibungsmerkmale zu finden, anhand derer Wiederholungen im Stück gefunden werden können. Diese können rhythmischer, spektraler (Klangfarbe) oder musikalischer (Tonhöhe bzw. Harmonie) Natur sein.

3.1 Rhythmische Struktur

3.1.1 Rhythmische Struktur mittels HMM

Die rhythmische Struktur eines Musikstückes kann verwendet werden, um Wiederholungen bezüglich des *timings*, sprich der Wiederholung gewisser rhythmischer Muster, zu finden. Es kann zum Beispiel sein, dass im Refrain *half time* und in der Strophe *full time*[1] gespielt wird. Weiters kann es sein, dass einem Teil des Stücks eine *Sechzehntel*-Struktur zugrunde liegt, während in einem anderen Teil haupsächlich *Achtel*-Noten gespielt werden.

Für die rythmische Struktur können drei Einheiten definiert werden:

- *Tatum*: Kleinste zeitliche Einheit, in der Änderungen, bzw. rhythmische Einheiten stattfinden (*temporal atom*)

[1] *half time* bedeutet, dass bei gleichbleibendem Tempo die Zählzeit halbiert wird

- *Tactus / beat*: Tempo des Stückes, normalerweise ganzahliges Vielfaches des *Tatum*
- *Measure*: Akkordwechselrate oder Länge des Rhythmuspatterns (Vielfaches von *Tactus*)

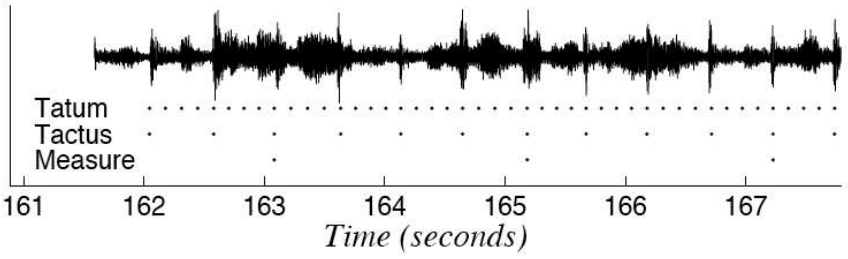

Abbildung 3.1: *Tatum*, *Tactus* und *Measure* (vgl. [15])

Als zeitliche Atome können zum Beispiel Sechzehntelnoten gesehen werden, der Tactus (Beat) steht normalerweise für die Viertelnoten (beziehungsweise das *Grundtempo*) und Measure kann eine Taktlänge oder eine Akkordwechselrate angeben.

In [14, 15] wird eine Methode zum Detektieren dieser verschiedenen Maße in einem Stück vorgestellt. Hierbei werden zwei Modelle kombiniert; das erste berechnet die Puls-Perioden der oben erläuterten Einheiten, das zweite die *Phase,* welche die *Startpunkte* für die einzelnen Einheiten markiert. Beiden Modellen geht eine *Zeit-Frequenz-Analyse* und eine *Kammfilterbank* voraus.

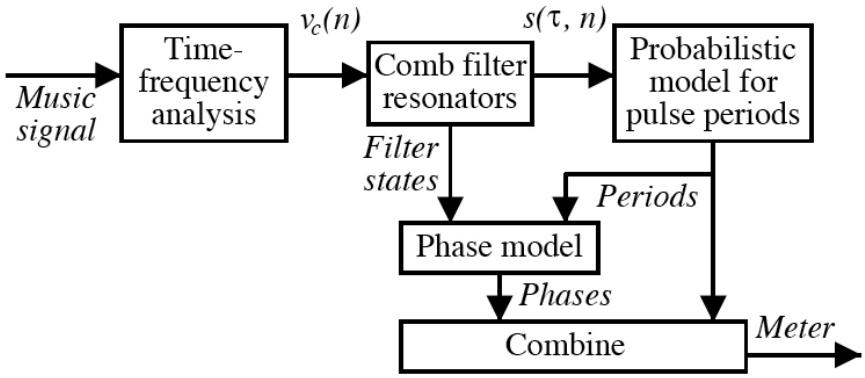

Abbildung 3.2: Blockschaltbild der Meter-Analyse (aus [15])

Zeit - Frequenz - Analyse

Das akustische Eingangssignal (16 bit @ 44,1 kHz) wird mittels einer blockweise Kurzzeit-Fourriertransformation in den Frequenzbereich transformiert. Die einzelnen überlappenden (50% *overlap*) Signalblöcke á 23ms (1024 Samples) werden vor der Transformation mit einem Hannigfenster gewichtet. Jeder transformierte Signalblock wird im Frequenzbereich zu jeweils 36 überlappenden (energieäquivalenten) kritischen Bändern (CB) zusammengefasst. Die Energie eines jeden Filterausgangs wird gespeichert und es ergibt sich pro Frame ein Vektor $x_b(k)$, wobei k der Zeitindex und $b = 1, 2, \ldots, 36$ die Nummer des jeweiligen Subbandes ist.

Um den Grad der Änderung in der Leistung zu berechnen, wird die μ-*law*-Kompression herangezogen. Diese basiert im Prinzip auf dem Weber-Frechner'schen Gesetz, das besagt, dass $\frac{\Delta I}{I}$ von 20 dB bis 100 dB ungefähr konstant ist[2]. Das bedeutet, die kleinste wahrmehmbare Änderung der Intensität ist proportional zur jeweiligen Lautstärke. Die μ-*law*-Kompression stellt eine flexible Form zur Berechnung dar:

$$y_b(k) = \frac{ln(1 + \mu x_b(k))}{ln(1 + \mu)} \tag{3.1}$$

wobei mit dem Wert für μ zwischen linearer (kleiner Wert) und logarithmischer (großer Wert) Transformation gewählt werden kann.

Das erhaltene Signal wird nun für eine bessere zeitliche Auflösung durch Einfügen von Nullen um den Faktor zwei upgesampled, was zu einer Samplingrate von f_r =172Hz führt. Auf dieses Signal wird ein Butterworth-Filter sechster Ordnung mit *cut-off*-Frequenz f_{LP} =10 Hz angewandt, welches die Gruppenlaufzeiten des LP-Filters durch Vorwärts- und Rückwärtsfilterung kompensiert. Man erhält wiederum ein geglättetes, stetiges aber bandbreitenreduziertes Signal. Dieses wird mit $z_b(n)$ bezeichnet. Dieses Signal wird anschließend noch durch einen Halbweg-Gleichrichter (HWR) gleichgerichtet, um die nachfolgende Differentiation sinnvoll zu machen:

[2]Die Lautheitsempfindung liese sich besser durch das Potenzgesetz $\left(\frac{\Delta I}{I}\right)^3$nach Stevens beschreiben, welches besser mit den Empfindungsdaten übereinstimmt.

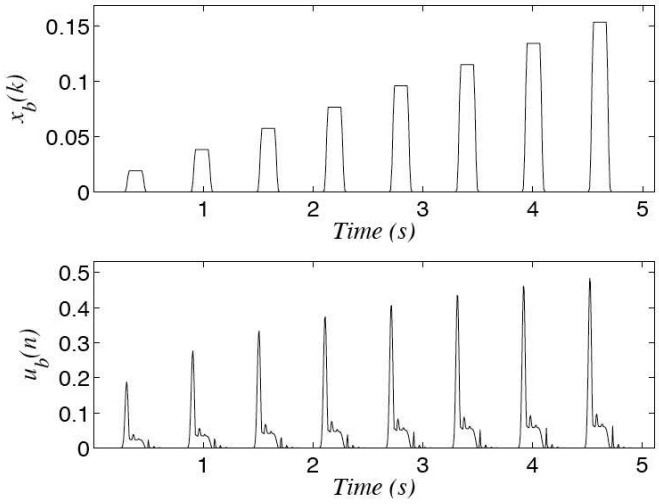

Abbildung 3.3: Gefiltertes Eingangssignal $x_b(k)$ und dynamisch komprimierte Differentiation $u_b(n)$. Man erkennt deutlich die Spitzen bei starker Änderung der Dynamik. (vgl. [15])

$$z'_b = HWR(z_b(n) - z_b(n-1)). \tag{3.2}$$

Aus $z_b(n)$ und $z'_b(n)$ wird dann die Differentiation folgendermaßen berechnet:

$$u_b(n) = (1-\lambda)z_b(n) + \lambda\frac{f_r}{f_{LP}}z'_b(n). \tag{3.3}$$

λ bestimmt die Balance zwischen $z_b(n)$ und $z'_b(n)$ und nimmt einen Wert zwischen null und eins an (0.8 wurde verwendet). Der Faktor $\frac{f_r}{f_{LP}}$ kompensiert die kleine Amplitude des differenzierten Signals. Nun werden jeweils m_0 benachbarte Frequenzbänder aufaddiert und es bleiben $c_0 = \lceil b_0/m_0 \rceil$ [3] Bänder übrig:

$$v_c(n) = \sum_{b=(c-1)m_0+1}^{cm_0} u_b(n), \quad c = 1, \ldots c_0. \tag{3.4}$$

Diese Signale repräsentieren den Grad der Akzentuierung des Musikstückes als Funktion der Zeit.

[3] $\lceil \ldots \rceil$ bedeutet, dass auf die nächste ganze Zahl aufgerundet wird

Um nun in $v_c(n)$ eine Periodizität zu finden wird eine Bank von Kammfiltern mit unterschiedlichen Resonanzfrequenzen und jeweils konstanter Abklingkonstante (bzw. Halbwertszeit) verwendet. Die Halbwertszeit beschreibt jene Zeitspanne die vergeht, bis das eingeschwungene Filterausgangssignal ohne weitere Energiezuführung am Eingang des (Kamm)-Filters auf den halben Ausgangswert abfällt. Der Ausgang des Kammfilters für die Verzögerung τ und Eingang $v_c(n)$ ergibt sich aus:

$$r_c(\tau, n) = \alpha_\tau r_c(\tau, n-\tau) + (1-\alpha_\tau) v_c(n) \tag{3.5}$$

mit $\alpha_\tau = 0.5^{\frac{\tau}{T_0}}$. Als Halbwertszeit werden dabei drei Sekunden verwendet, also $T_0 = 3 f_r$ was 688 Samples entspricht.

Eine solche Bank von Kammfiltern wird auf das Signal angewandt, wobei für die einzelnen Kammfilter der Filterbank die Werte τ von 1 bis 688 zum Einsatz kommen. Die Momentanenergie wird aus

$$\hat{r}_c(\tau, n) = \frac{1}{\tau} \sum_{i=n-\tau+1}^{n} r_c(\tau, i)^2 \tag{3.6}$$

berechnet. Die Energie wird mit der Gesamtenergie

$$\gamma(\alpha_\tau) = \frac{(1-\alpha_\tau)^2}{1-\alpha_\tau^2} \tag{3.7}$$

normalisiert, um Unterschieden in den verschiedenen Antworten, welche durch verschiedene α_τ entstehen, Rechnung zu tragen.

$$s_c(\tau, n) = \frac{1}{1-\gamma(\alpha_\tau)} \left(\frac{\hat{r}_c(\tau, n)}{\hat{v}_c(n)} - \gamma(\alpha_\tau) \right) \tag{3.8}$$

$\hat{v}_c(n)$ wird dabei mittels eines Integrators mit Halbwertszeiten von drei Sekunden und $\tau = 1$ berechnet. Diese Normalisierung ist vorteilhaft, weil dadurch für alle Peak-Frequenzen ein Einheitspuls stehenbleibt und der von τ abhängige Trend weitgehend eliminiert werden kann (siehe Abbildung 3.4).

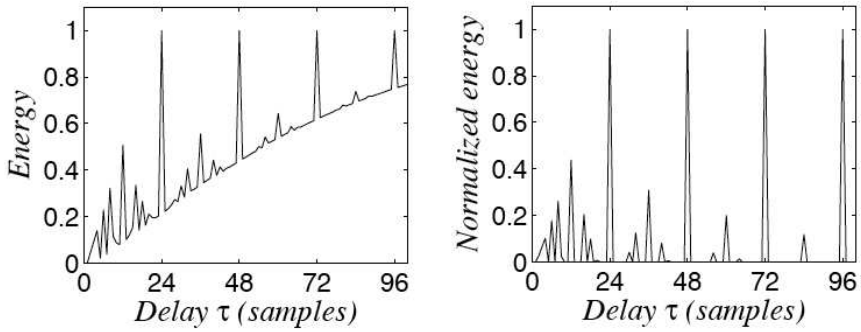

Abbildung 3.4: Resonator Energie $\hat{r}_c(\tau, n)$ und normalisierte Energie $s_c(\tau, n)$ für einen Impulszug mit $\tau = 24$ Samples [15].

Schlussendlich werden die vier Bänder aufsummiert um ein einziges Signal $s_c(\tau, n)$ zu erhalten, welches für das folgende Wahrscheinlichkeitsmodell als Beobachtung dient:

$$s(\tau, n) = \sum_{c=1}^{c_0} s_c(\tau, n) \tag{3.9}$$

Für die *Tatum*-Schätzung wird ein Leistungsspektrum wie folgt berechnet:

$$S(f, n) = f \mid \frac{1}{\tau_{max}} \sum_{\tau=1}^{\tau_{max}} \left(s(\tau, n)\zeta(\tau)e^{-j2\pi f(\tau-1)/\tau_{max}} \right) \mid^2 \tag{3.10}$$

wobei f den spektralen Trend kompensiert und $\zeta(\tau)$ ein halbes Hanningfenster (siehe Abbildung 3.5) ist.

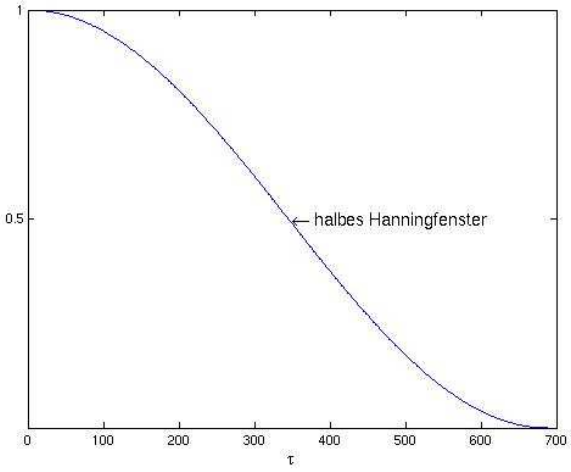

Abbildung 3.5: Halbes Hanningfenster zur Berechnung von $S(f, n)$

Puls - Perioden - Modell

Das in [14, 15] vorgestellte Modell schätzt zuerst die Periode und erst dann den Zeitpunkt in der Signalfolge (Phase), da die Periodendauern unabhängig von der Phase berechnet werden können. Da das Tempo innerhalb eines Musikstückes nicht zwangsläufig konstant sein muss, sollte das mathematische Modell auf gewisse Änderungen reagieren beziehungsweise solche möglichen Änderungen miteinbeziehen. Auch die Zusammenhänge zwischen den einzelnen metrischen Maßen müssen berücksichtigt werden. Dazu wird ein *Hidden-Markov-Modell* mit drei verborgenen Zuständen verwendet.

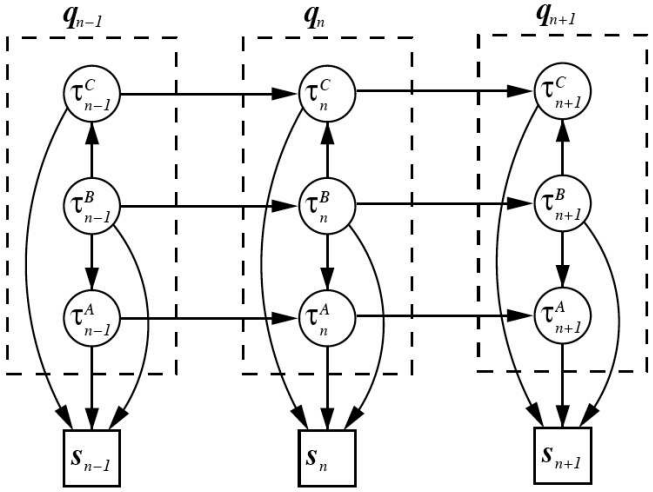

Abbildung 3.6: Hidden - Markov - Modell für die Schätzung von *tatum*, *tactus* und *measure* [15]

Als Beobachtung dient dabei das Signal $s(\tau, n)$, im Folgenden mit s_n bezeichnet. Die verborgenen Zustände sind dabei die *Tatum*-Periode τ_n^A, die *Beat*-Periode τ_n^B und die *Measure*-Periode τ_n^C. Um die Schreibweise zu vereinfachen wird in der folgenden Beschreibung der Modelle ein Zustand als $q_n = [j,\ k,\ l] \hat{=} [\tau_n^A,\ \tau_n^B,\ \tau_n^C]$ geschrieben. Das Hidden-Markov-Modell hat eine anfängliche Zustandsverteilung $P(q_1)$, Übergangswahrscheinlichkeiten $P(q_n \mid q_{n-1})$ und eine Wahrscheinlichkeitsdichtefunktion für die Beobachtungen $p(s_n \mid q_n)$, die nur vom jeweiligen Zustand abhängt. Somit kann die bedingte Wahrscheinlichkeit einer Zustandssequenz $Q = (q_1 q_2 \dots q_N)$ und einer Beobachtungssequenz $O = (s_1 s_2 \dots s_N)$ als

$$p(Q, O) = P(q_1)p(s_1 \mid q_1) \prod_{n=2}^{N} P(q_n \mid q_{n-1})p(s_n \mid q_n) \tag{3.11}$$

geschrieben werden. Um die Übergangswahrscheinlichkeit zu vereinfachen, kann angenommen werden, dass für die Schätzung des Beat - Maßes τ_n^B nur das vorangegangene Beat - Maß τ_{n-1}^B berücksichtigt werden muss. Für das Tatum - und Measure - Maß (τ_n^A,τ_n^C) gilt die Annahme, dass nur das vorangegangene gleiche Maß (τ_{n-1}^Abeziehungsweise τ_{n-1}^C) und das zum jeweiligen Zeitpunkt beobachtete τ_n^B für die Schätzung relevant ist. Daraus ergibt sich die Übergangswahrscheinlichkeit zu:

$$P(q_n \mid q_{n-1}) = P(\tau_n^B \mid \tau_{n-1}^B)P(\tau_n^A \mid \tau_n^B, \tau_{n-1}^A)P(\tau_n^C \mid \tau_n^B, \tau_{n-1}^C). \tag{3.12}$$

Von den selben Annahmen ausgehend kann man für die Anfangswahrscheinlichkeit folgende Gleichung aufstellen:

$$P(q_1) = P(\tau_1^B)P(\tau_1^A \mid \tau_1^B)P(\tau_1^C \mid \tau_1^B). \tag{3.13}$$

Nun gilt es, für die in den obigen Gleichungen vorkommenden Modellparameter sinnvolle Schätzwerte zu berechnen. Aus einer Serie von in [14, 15] angestellten Annahmen kann die Wahrscheinlichkeitsdichtefuntkion für die Beobachtungen ganz einfach aus $s(\tau, n)$ und $S(f, n)$ zu jedem Zeitpunkt wie folgt berechnet werden:

$$p(s_n \mid q_n) \propto s(k)s(l)S(1/j). \tag{3.14}$$

Die Übergangswahrscheinlichkeiten $P(\tau_n^i \mid \tau_{n-1}^i)$, $i \in \{A, B, C\}$ ergeben sich aus der jeweiligen Anfangswahrscheinlichkeit $P(\tau_1^i)$ und einer Funktion $f(\tau_n^i/\tau_{n-1}^i)$, welche die Änderungen beziehungsweise Schwankungen im Tempo berücksichtigt:

$$P(\tau_n^i \mid \tau_{n-1}^i) = P(\tau_1^i) f\left(\frac{\tau_n^i}{\tau_{n-1}^i}\right). \tag{3.15}$$

$f(\tau_n^i/\tau_{n-1}^i)$ implementiert dabei eine Normalverteilung als Funktion des Logarithmus zweier aufeinanderfolgenden Periodenwerte:

$$f\left(\frac{\tau_n^i}{\tau_{n-1}^i}\right) = \frac{1}{\sigma_1\sqrt{2\pi}} exp\left[-\frac{1}{2\sigma_1^2}\left(\ln\left(\frac{\tau_n^i}{\tau_{n-1}^i}\right)\right)^2\right] \tag{3.16}$$

mit $\sigma_1 = 0.2$.

Für die Verteilung der verschiedenen Tempomaße wird für jedes der Maße eine logarithmische Normalverteilung mit verschiedenen Parametern angewandt, welche für eine allgemeine Beschreibung dient.

$$p(\tau^i) = \frac{1}{\tau^i\sigma_1\sqrt{2\pi}} exp\left[-\frac{1}{2(\sigma^i)^2}\left(\ln\left(\frac{\tau^i}{m^i}\right)\right)^2\right] \tag{3.17}$$

Maß	σ	m
tatum	*0.39*	*0.18*
beat	*0.28*	*0.55*
meaure	*0.26*	*2.1*

Tabelle 3.1: Mittelwert und Standardabweichung für die verschiedenen Maße

Folgende Abbildung zeigt die Verteilungen der drei Maße:

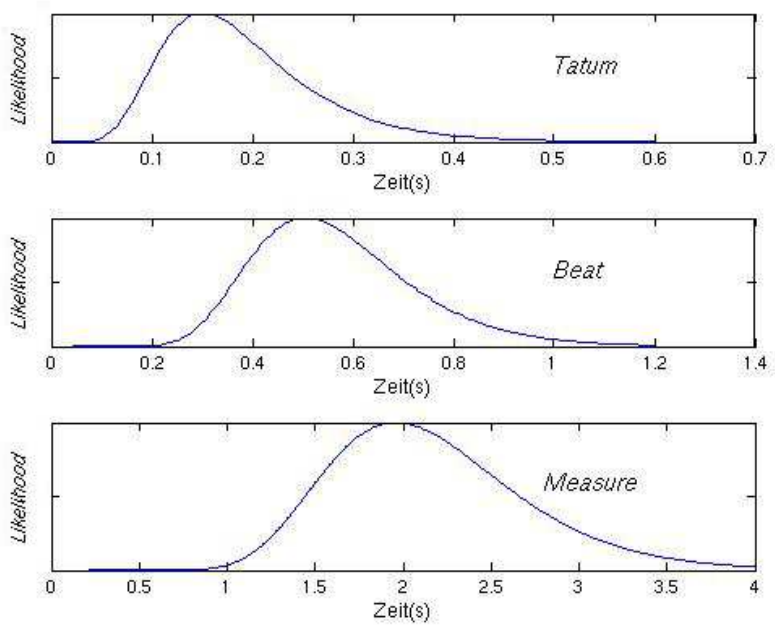

Abbildung 3.7: Verteilung für das *tatum*-, *beat*- und *measure*-Maß

Die entsprechenden Werte für σ^i und m^i wurden dabei aus [15] entnommen und sind in Tabelle (3.1) aufgelistet.

Der Zusammenhang zwischen gleichzeitig auftretenden Tempo-Werten ergeben sich aus superponierten Gauß'schen Verteilungen, d.h. einem so genannten *Gaussian Mixture Model*, das einen Schwerpunkt auf binäre oder ternere Zusammenhänge legt:

$$g\left(\frac{\tau^i}{\tau^j}\right) = \sum_{l=1}^{9} w_l N\left(\frac{\tau^i}{\tau^j}; l, \sigma_2\right) \tag{3.18}$$

Dabei sind w_l die Gewichte der einzelnen Komponenten und $N(x)$ steht für eine

Gauß'sche verteilung. σ_2 ist auf 0.3 festgelegt und der Parameter l bestimmt jeweils den Mittelwert und damit den *offset* der jeweiligen Normalverteilung.

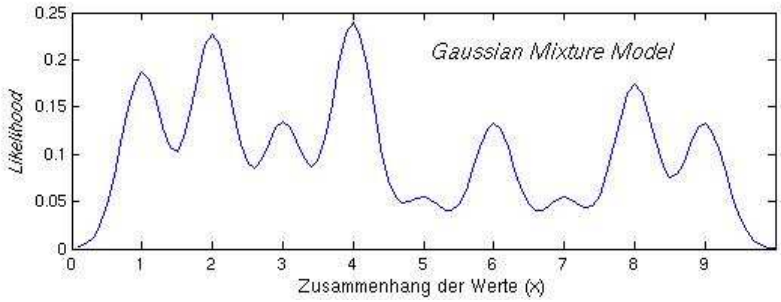

Abbildung 3.8: Verteilung $g(x)$ (Gleichung 3.18), die den Zusammenhang gleichzeitig auftretender Maße modelliert

Insgesamt ergibt sich dadurch die Übergangswahrscheinlichkeit zu:

$$P(q_n \mid q_{n-1}) = P(\tau_1^B) f\left(\frac{\tau_n^B}{\tau_{n-1}^B}\right) P(\tau_1^A) f\left(\frac{\tau_n^A}{\tau_{n-1}^A}\right) g\left(\frac{\tau_n^B}{\tau_n^A}\right) P(\tau_1^C) f\left(\frac{\tau_n^C}{\tau_{n-1}^C}\right) g\left(\frac{\tau_n^C}{\tau_n^B}\right) \tag{3.19}$$

Nun gilt es, die wahrscheinlichste Sequenz $Q = (q_1, q_2, \ldots, q_N)$ der Zustandsvariablen bei gegebenen Beobachtungen $O = (s_1, s_2, \ldots, s_N)$ zu finden. Dazu werden pro Sekunde die fünf besten Kandidaten für τ^A,τ^Bund τ^Causgewählt. Diese Auswahl wird getroffen, indem in jeder Sekunde $p(\tau_n^i)p(s_n \mid \tau_n^i)$ für $i \in \{A, B, C\}$maximiert wird. Somit ergeben sich für die drei Maße 125 mögliche Kombinationen, für die nun die Wahrscheinlichkeiten berechnet werden müssen. Weiters müssen 125 x 125 Übergangswahrscheinlichkeiten bestimmt werden. Diese errechneten Daten werden nun für den darauf folgenden Viterbi-Algorithmus verwendet. Dieser Algorithmus dient dazu, den wahrscheinlichsten Weg, sprich die wahrscheinlichste Zustandsfolge zu jeder Beobachtung zu finden. Dazu beginnt man von hinten, also von der letzten Beobachtung, und sucht sich nach vorne durch., indem man jeweils die größte Übergangswahrscheinlichkeit auf dem Weg nach vorne wählt. Man erhält also 125 Wege vom Ende der Sequenz bis zum Anfang. Nun geht man von vorne den besten dieser Wege nach, um den wahrscheinlichsten Pfad durch alle Beobachtungen zu fin-

den. Dieser Pfad, bzw. die Zustände entlang dieses Pfades sind dann die geschätzten Werte für die drei Maße zu jedem Zeitpunkt.

Phasen - Modell[4]

Nachdem die Puls-Perioden für jeden Zeitpunkt ausgewählt wurden, kann man die Phase schätzen. Die Phase ist dabei der zeitliche Anker, sprich jener Punkt nach dem Beginn der aktuellen Sekunde, an dem die rhythmische Einheit zum ersten Mal auftritt. Durch die Perioden und die Phase kann die rhythmische Struktur eines Stückes vollständig beschrieben werden. Es ist jedoch ausreichend, nur für *beat* und *measure* die Phase zu schätzen, da die Phase φ^Bdes *beats* auch für φ^A, die Phase des rhythmischen Atoms verwendet werden kann. Somit können die beiden Phasen mittels zweier unabhängiger Hidden-Markov-Modellen geschätzt werden. Als Beobachtungen für die Modelle wird der Ausgang der Resonatoren $r_c(\hat{\tau}_n^i, j)$ wobei j jeweils innerhalb des Beobachtungsfensters einer Periode (d.h j läuft von $n-\tau+1$ bis n) liegt und $\hat{\tau}_n^i$ für die gefundene Periode steht. Um im Folgenden die Schreibweise zu vereinfachen, wird der betreffende Ausgang der Resonatoren mit R_n^i bezeichnet und die einzelnen Elemente dieser Matrix mit $(R_n^i)_{c,j}$ (siehe rechteckiger Kasten in Abbildung 3.9).

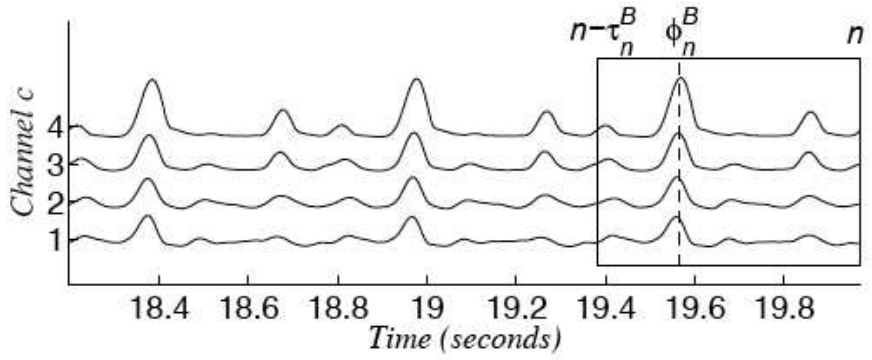

Abbildung 3.9: Phase ϕ_n^B für die Matrix R_n^B (rechteckiger Kasten)

[4]Phase steht in diesem Zusammenhang für den zeitpunkt des *beats* in der Signalfolge

Die verborgene Variable in beiden Modellen ist die zu bestimmende Phase, welche Werte zwischen $n - \hat{\tau}_n^i + 1$ und n einnehmen kann. Das Modell hat wiederum eine Anfangswahrscheinlichkeit $P(\varphi_1)$, Übergangswahrscheinlichkeiten $P(\varphi_n \mid \varphi_{n-1})$ und die Auftrittswahrscheinlichkeit $p(R_n^i \mid \varphi_n^i)$, welche nur vom aktuellen Zustand abhängt und sich für die *beat*-Phase wie folgt berechnet:

$$p(R_n^B \mid \varphi_n^B = j) \propto \sum_{c=1}^{c_0} (c_0 - c + 2)(R_n^B)_{c,j} \tag{3.20}$$

wobei $c = 1$ für den tiefsten Frequenzkanal steht. Die Auftrittswahrscheinlichkeit ist also eine gewichtete Summe der Resonatoren, wobei die tiefen Frequenzen stärker ins Gewicht fallen, da man von einem stabilen Bass ausgeht, das heißt, man geht davon aus, dass der Bass grundlegend den Rhythmus bestimmt.

Für die Bestimmung der *measure*-Phase ist eine etwas komplexere Berechnung erforderlich. Zuerst wird ein Vektor

$$h_n(l) = \sum_{c=1}^{c_0} \sum_{k=0}^{3} \eta_{c,k} (R_n^C)_{c,j(k,l,n)} \tag{3.21}$$

mit $l = 0, 1, \ldots, \hat{\tau}_n^C - 1$ und

$$j(k, l, n) = n - \hat{\tau}_n^C + 1 + \left(\left(l + \frac{k\hat{\tau}_n^C}{4} \right) mod \hat{\tau}_n^C \right). \tag{3.22}$$

Somit ergibt sich mit dem *delay* $k = 0, ..., 3$ eine Unterteilung von $\hat{\tau}_n^C$ in vier Teile. Mittels zweier Matritzen $\eta_{n,c}^{(1,2)}$, die unterschiedliche Gewichte für die Verzögerungen in den verschiedenen Kanälen c enthalten, werden zwei verschiedene Vektoren $h_n^{(1)}(l)$ und $h_n^{(2)}(l)$ berechnet. Einer steht dabei für eine Unterteilung des *measure*-Maßes, der zweite für eine Halbierung. Aus diesen beiden Vektoren wird nun jeweils das Maximum genommen und man erhält einen einzigen Vektor

$$h_n^{(1,2)}(l) = \max \left(h_n^1(l), h_n^2(l) \right). \tag{3.23}$$

Mit Hilfe dieses Vektors kann nun die Wahrscheinlichkeit der Beobachtung wie

folgt berechnet werden:

$$p(R_n^C \mid \varphi_n^C = j) \propto h_n^{(1,2)}(j - (n - \hat{\tau}_n^C + 1)) \tag{3.24}$$

Die Übergangswahrscheinlichkeiten können als Funktion des Vorhersagefehlers berechnet werden. Dieser Fehler misst die Abweichung der aktuellen Phase φ_n^ivon der nächsten rhythmischen Einheit, welche aus der vorhergehenden Phase φ_{n-1}^iund dem der Puls-Periode $\hat{\tau}_n^i$ errechnet wird:

$$e = \frac{1}{\hat{\tau}_n^i}\left\{\left[\left(\mid \varphi_n^i - \varphi_{n-1}^i \mid + \frac{\hat{\tau}_n^i}{2}\right) mod \hat{\tau}_n^i\right] - \frac{\hat{\tau}_n^i}{2}\right\} \tag{3.25}$$

Die Übergangswahrscheinlichkeit ergibt sich damit zu:

$$P(\varphi_n^i \mid \varphi_{n-1}^i) = \frac{1}{\sigma_3\sqrt{2\pi}} exp\left(-\frac{e^2}{2\sigma_3^2}\right) \tag{3.26}$$

wobei $\sigma_3 = 0.1$ für *beat* und *measure* zum Einsatz kommt. Die Phasen werden Anfangs als gleichverteilt angenommen. Fünfzehn beste Kandidaten werden pro Sekunde extrahiert und die beiden Modelle gelöst. So kann für das Stück zu jedem Zeitpunkt das Tempo und die Position für die rhythmische Struktur festgelegt werden.

3.1.2 Rhythmische Struktur mittels onset detection

Ausgehend von einer *onset-detection* kann auf das Tempo geschlossen werden. Dazu wird ein *two-state*-Modell [9] verwendet, welches zuerst ohne jedes Vorwissen über das Eingangssignal eine *beat-detection* vornimmt und dann, ausgehend vom ersten, generellen Zustand einen vom musikalischen Kontext abhängigen Zustand berechnet. Es werden jeweils das Tempo und die Position der *beats* detektiert. Die *onset*-Funktion wird dabei im Spektralbereich (vgl. Kapitel 2.5) berechnet:

$$\Gamma(m) = \sum_{k=1}^{K} \mid X_k(m) - \hat{X}_k(m) \mid^2 \tag{3.27}$$

wobei $X_k(m)$ das tatsächliche Spektrum und $\hat{X}_k(m)$ der aus den vorangegangene Frames geschätzte Wert für das komplexe Spektrum ist, wobei laut [9] der Gleichanteil nicht berücksichtigt werden muss $(k = 1, ...K)$. Die zeitliche Auflösung beträgt dabei $t_{DF} = 11.6$ms. Die so erhaltene Funktion wird in überlappende Frames unterteilt, um Tempo- und Phasenänderungen zuzulassen:

$$\Gamma_i(m) = \begin{cases} \Gamma(m), & m = 1 + (i-1)B_h, \ldots, B_f + (i-1)B_h \\ 0, & sonst \end{cases} \tag{3.28}$$

mit Framelänge $B_f = 512$DF (*detection-function*) Samples und Sprungweite $B_h = B_f/4$, was einer Fensterlänge von 6 Sekunden bei einer Überlappung von 75% entspricht. Ausgehend von diesem Framing wird nun zum einen der *general state* und aus diesem der *context dependent state* berechnet.

General State

Im *general state* werden Position und Periodendauer ohne jegliches Vorwissen zu jedem Zeitpunkt berechnet. Dazu wird als erstes die Periodendauer für das Tempo berechnet und erst dann die Position der Schläge bestimmt. Die Phaseninformation wird durch die Verwendung einer Autokorrelationsfunktion verworfen. Für diese Funktion ein adaptiver Schwellwert

$$\bar{\Gamma}_i(m) = \text{mean}\{\Gamma_i(q)\} \quad m - \frac{Q}{2} \leq q \leq m + \frac{Q}{2} \tag{3.29}$$

mit $Q = 16$DF Samples berechnet. Dieser wird dann vom originalen Wert der *detection-function* abgezogen und die so erhaltene neue Funktion wird gleichgerichtet:

$$\tilde{\Gamma}_i(m) = \text{HWR}(\Gamma_i(m) - \bar{\Gamma}_i(m)) \tag{3.30}$$

wobei $\text{HWR}(x) = (x + \mid x \mid)/2$. Nun kann die Autokorrelationsfolge wie folgt berechnet werden:

$$A(l) = \frac{\sum_{m=1}^{B_f} \tilde{\Gamma}_i(m)\tilde{\Gamma}_i(m-l)}{\mid l - B_f \mid} \quad l = 1, \ldots, B_f. \tag{3.31}$$

Um nun Periodizitäten in der Autokorrelation zu finden, wird eine Kammfiltervorlage mit Periodendauern von $\tau = 1$ bis $\tau_{max} = B_f$ verwendet:

$$\lambda_\tau(l) = \sum_{p=1}^{4} \sum_{v=1-p}^{p-1} \frac{\delta(l - \tau p + v)}{2p - 1} \quad l = 1, \ldots, B_f. \tag{3.32}$$

Dies erlaubt die Verwendung von vier solchen Kammfilterelementen innerhalb einer Vorlage für ein Kammfilter. Um die schlechte zeitliche Auflösung für kurze Verzögerungen zu berücksichtigen hat jedes Kammfilterelement eine zu τ proportionale Breite, welche von $v = 1 - p, \ldots p - 1$ bestimmt ist und wird in der Höhe durch $2p - 1$ normalisiert.

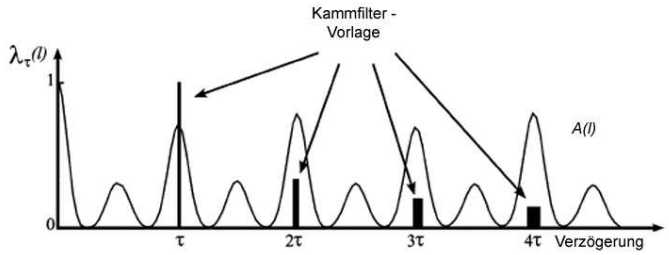

Abbildung 3.10: Kammfiltervorlage $\lambda_\tau(l)$ und Autokorrelationsmatrix $A(l)$ (vlg. [9])

Mehrere solcher Kammfilter werden nun in einer *shift-invarianten* Kammfilterbank $F_G(l, \tau)$ kombiniert und mittels einer Gewichtungskurve wird der Schwerpunkt für das Tempo auf 120 bpm (*beats per minute*) gelegt. Dazu wird eine Rayleigh - Verteilung

$$w_G(\tau) = \frac{\tau}{\beta^2} e^{-\tau^2/2\beta^2} \quad \tau = 1, \ldots, B_h \tag{3.33}$$

mit $\beta = 43$ verwendet. Die Gewichtungsmatrix ergibt sich dadurch zu:

$$F_G(l, \tau) = w_G(\tau)\lambda_\tau(l) \tag{3.34}$$

und kann somit in Matrix - From angeschreiben werden.

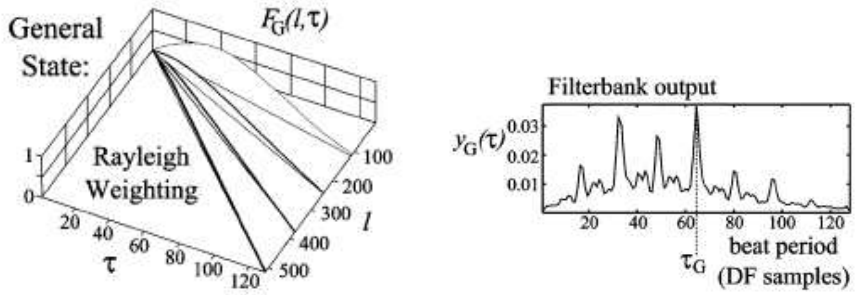

Abbildung 3.11: Kammfilterbank $F_G(l,\tau)$ für den *general state,* Ausgang der Filterbank $y_G(\tau)$ und τ_G als geschätzte *beat*-Periode (aus [9])

Mutlipliziert man diese Matrix mit der Autokorrelationsfunktion, so erhält man eine Funktion

$$y_G(\tau) = \sum_{l=1}^{B_f} A(l) F_G(l,\tau) \tag{3.35}$$

aus der die *beat*-Periode als Index des Maximalwertes extrahiert werden kann,

$$\tau_G = \arg\max_{\tau}(y_G(\tau)) \tag{3.36}$$

da hier alle Verzögerungen über das gesamte Frame aufsummiert werden.

Der nächste Schritt ist nun die Positionierung der *beats* innerhalb eines jeden Frames. Dazu kann entweder eine kausale oder eine nichtkausale Analyse verwendet werden. Zuerst wird die nichtkausale Berechnung gezeigt. Für die Ausrichtung der *beats* wird ein ähnlicher, matrizenbasierter Ansatz als für die Periodenberechnung verwendet. Es wird wiederum eine Kammfiltervorlage $\psi_\alpha(m)$ erstellt, aus welcher eine Kammfiltermatrix $H_G(m,\alpha)$ für die Positionierung berechnet wird. Jede der Kammfiltervorlagen wird mit einem linear abfallenden Gewicht $v(m)$ multipiziert:

$$\psi_\alpha(m) = \sum_{k=1}^{\lfloor B_f/\tau_G \rfloor} v(m)\delta(m - k\tau_G + \alpha) \quad m = 1,\ldots,B_f \tag{3.37}$$

wobei k über die Nummer aller Elemente in jeder Vorlage reicht und $v(m)$ sich wie

folgt berechnet:

$$v(m) = \frac{B_f - m}{B_f}. \tag{3.38}$$

In jeder Spalte der Matrix $H_G(m, \alpha)$ steht nun eine Kammfiltervorlage mit jeweils größer werdendem Offset α

$$H_G(m, \alpha) = \psi_\alpha(m) \quad 1 \leq \alpha \leq \tau_G. \tag{3.39}$$

Für die nichtkausale Analyse ergibt sich aus dem Produkt des *detection-function* Frames $\Gamma_i(m)$ und $H_G(m, \alpha)$ eine Funktion

$$z_G(\alpha) = \sum_{m=1}^{B_f} \Gamma_i(m) H_G(m, \alpha), \tag{3.40}$$

aus der der Abstand α_G zum Anfang des aktuellen Frames aus

$$\alpha_G = \arg\max_{\alpha} z_G(\alpha) \tag{3.41}$$

berechnet werden kann.

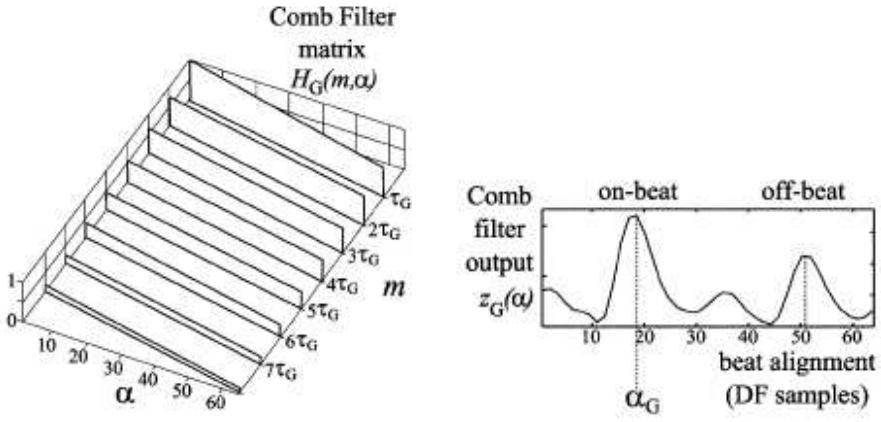

Abbildung 3.12: Kammfiltervorlage $H_G(m, \alpha)$, Ausgang $z_G(\alpha)$ und *beat* - Position α_G für den *general state* (vgl. [9])

Nun werden *beats* $\gamma_{i,b}$in Intevallen der *beat*-Periode im Abstand von α_G zum

Startzeitpunkt des Frames t_i plaziert:

$$\gamma_{i,b} = (t_i + \alpha_G) + (b-1)\tau_G \tag{3.42}$$

mit der Beschränkung $\gamma_{i,b} < t_i + B_h$.

Für die kausale Version wird $\Gamma_i(m)$ durch seine Zeitinverse $\Gamma_i^*(m)$ ersetzt und mit der selben Matrix $H_G(m,\alpha)$ multipliziert, um die Kammfilterelemente am Ende des Frames stärker zu betonen. Die kausale Version von $z_G(\alpha)$ ergibt sich zu

$$z_G^*(\alpha) = \sum_{m=1}^{B_f} \Gamma_i^*(m) H_G(m,\alpha). \tag{3.43}$$

Analog zu nichtkausalen Berechnung können die *beat*-Positionen $\gamma_{i,b}^*$ über das Ende t_i^*des i -ten Frames hinaus vorhergesagt werden:

$$\gamma_{i,b}^* = (t_i^* + \alpha_G^*) + b\tau_G \tag{3.44}$$

α_G^*wird dabei gleich wie oben als

$$\alpha_G^* = \arg\max_{\alpha} z_G^*(\alpha) \tag{3.45}$$

errechnet. Für $\gamma_{i,b}^*$ gilt wieder die Restriktion $\gamma_{i,B}^* < t_i^* + B_h$.

Context - Dependent State

Die im *general state* erhaltenen Werte für $\tau_G$und $\alpha_G$enthalten keine Information über den vergangenen Verlauf und deshalb kann nicht auf die Kontinuität des *beats* geschlossen werden. Es können zwei grundsätzliche Fehler auftreten:

1. Es kann das doppelte oder halbe Tempo detektiert werden
2. Es kann eine Verschiebung von *on-beat* auf den *off-beat*[5] geschehen

[5] *on-beat* bedeutet, dass der *beat* genau auf den Viertelnoten plaziert wird, *off-beat* genau in den Stellen dazwischen, den (unbetonten) Achtelnoten

Um diesen Fehlern vorzubeugen wird der *context-dependent state* eingeführt, welcher ähnlich dem *general state* arbeitet, jedoch auch die schon berechneten Werte für *beat*-Periode und *beat*-Position miteinbezieht. Für die Berechnung von τ_C wird eine neue Kammfiltermatrix $F_C(l,\tau)$ berechnet. Dazu wird zuerst bestimmt, ob es sich um eine binäre (2/4, 4/4,...) oder ternere (3/4, 6/8,...) Taktart handelt [5]. Dazu wird die im vorigen erhaltene Autokorrelationsfunktion betrachtet und die Einstellung T als binär ($T = 4$) oder terner ($T = 3$) festgelegt:

$$T = \begin{cases} 4, & A(2\tau_G) + A(4\tau_G) > A(3\tau_G) + A(6\tau_G) \\ 3 & onst \end{cases}. \tag{3.46}$$

Dieser Wert T dient nun zur Berechnung der Kammfiltervorlage

$$\lambda_\tau(l) = \sum_{p=1}^{T} \sum_{v=1-p}^{p-1} \delta(l - \tau p + v) \quad l = 1, \ldots, B_f. \tag{3.47}$$

Das Wissen um die vorangegangenen Werte τ_G kann die *beat*-Periode anhand einer Gaußverteilung[6] auf ähnliche Werte bergenzt werden und die Kammfilterbank kann wie folgt berechnet werden:

$$F_C(l,\tau) = w_C(\tau)\lambda_\tau(l) \tag{3.48}$$

wobei für $w_C(\tau)$ eine Gauß'sche Verteilung um τ_G mit einer Standardabweichung von $\sigma_w = 4$ DF Samples eingesetzt wird:

$$w_C(\tau) = e^{-(\tau-\tau_G)^2/\sigma_w^2}. \tag{3.49}$$

Wie im *general state* wird das Maximum aus dem Produkt

$$y_C(\tau) = \sum_{l=1}^{B_f} A(l)F_C(l,\tau) \tag{3.50}$$

[6] wesentlich stärkere Konzentration um aktuellen Wert als bei Rayleigh-Verteilung

als *beat*-Periode verwendet:

$$\tau_C = \arg\max_{\tau}(y_C(\tau)). \tag{3.51}$$

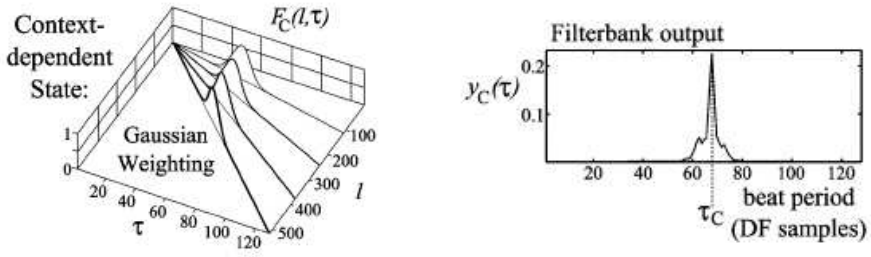

Abbildung 3.13: Filterbank $F_C(l,\tau)$ und Filterbankausgang $y_C(\tau)$ für den *context-dependent state* mit τ_C als *beat*-Periode (siehe [9])

Die Positionierung der *beats* im Frame erfolg ähnlich dem *general state*. Ausgehend von der gefundenen Verzögerung τ_C wird ein matrizenbasierter Ansatz verwendet, um die Startposition α_C im Frame zu finden. Auch hier kann wieder ein kausaler und ein nichtkausaler Ansatz gewählt werden. Man könnte zwar α_C aus dem letzten *beat* $\gamma_{i-1,B}$ des vorangegangenen Frames $\Gamma_{i-1}(m)$ und τ_C berechnen:

$$\alpha_C \approx \gamma_{i-1,B} + \tau_C \tag{3.52}$$

jedoch würde diese Berechnung auch nicht die geringste Schwankung in der Position des *beats* erlauben. Um solche kleinen Schwankungen zuzulassen, wird eine Gewichtsfunktion $\rho_C(\alpha)$ eingeführt, die auf der wahrscheinlichsten Position zentriert wird:

$$\rho_C(\alpha) = e^{-(\alpha-(\gamma_{i-1,B}+\tau_C))^2/2\sigma_p^2} \quad \alpha = 1,\ldots,\tau_C. \tag{3.53}$$

σ_p wird dabei so gewählt, dass eine Verschiebung auf den $off-beat$ unmöglich gemacht wird, das heißt, Schwankungen um $\tau_C/2$ werden ausgeblendet. Es wird eine vom *beat* abhängige Standardabweichung von $\sigma_p = \tau_C/4$ DF Samples verwendet. Die kontextabhängige Kammfiltermatrix $H_C(m,\alpha)$ wird aufgestellt, indem in jeder

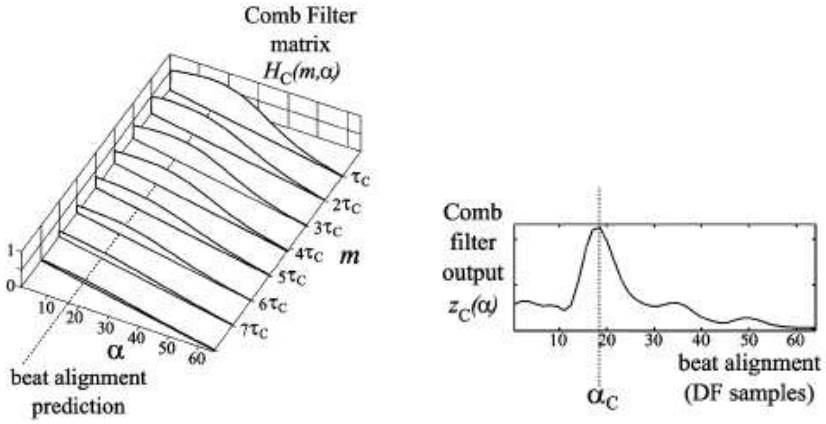

Abbildung 3.14: $H_C(m, \alpha)$ und dazugehöriger Ausgang $z_C(\alpha)$ mit α_C als *beat*-Position für den *context dependent state* [9]

Spalte ein mit $\rho_C(\alpha)$ gewichtetes Kammfilter $\psi_\alpha(m)$ steht:

$$\psi_\alpha(m) = \sum_{k=1}^{\lfloor B_f/\tau \rfloor} v(m)\delta(m - k\tau_C + \alpha) \quad m = 1, \ldots, B_f \tag{3.54}$$

mit $v(m)$ aus Gleichung 3.38. Die Kammfilterbank ergibt sich damit zu

$$H_C(m, \alpha) = \rho_C(\alpha)\psi_\alpha(m). \tag{3.55}$$

Die Startposition für den *beat* wird wiederum als Maximum von

$$z_C(\alpha) = \sum_{m=1}^{B_f} \Gamma_i(m) H_C(m, \alpha) \tag{3.56}$$

womit sich für den nichtkausalen Fall folgendes Ergebnis für α_C ergibt:

$$\alpha_C = \arg\max_\alpha z_C(\alpha) \tag{3.57}$$

Die Platzierung der *beat* erfolgt gleich wie für den *general state* (Gleichung 3.42). Für den kausalen Fall muss die Verteilungsfunktion geändert werden, da man hier vom Endpunkt des jeweiligen Frames ausgeht. Das bedeutet, dass α_C^* mit dem letzten *beat* des vorangegangenen Frames korrespondiert. Die Verteilungsfunktion ergibt

sich daher zu:

$$\rho_C^*(\alpha) = e^{-(\alpha-(\gamma^*_{i-1,B}))^2/2(\sigma_p^*)^2} \quad \alpha = 1, \ldots, \tau_C. \tag{3.58}$$

σ_p^* wird hier mit $\tau_C/8$ DF Samples festgelegt. Aus Gleichung 3.55 wird nun mit $\rho_C^*(\alpha)$ wieder eine Kammfiltermatrix berechnet:

$$H_C^*(m, \alpha) = \rho_C^*(\alpha)\psi_\alpha(m). \tag{3.59}$$

Das Produkt zwischen dem zeitinvertierten Frame $\Gamma_i^*(m)$ und $H_C^*(m, \alpha)$ wird

$$z_C^*(\alpha) = \sum_{m=1}^{B_f} \Gamma_i^*(m) H_C^*(m, \alpha) \tag{3.60}$$

berechnet, dessen Maximum α_C^* den Abstand des letzten beats in einem Frame vom Ende des Frames t_i^* angibt:

$$\alpha_C^* = \arg\max_\alpha z_C^*(\alpha). \tag{3.61}$$

Ähnlich wie beim *general state* 3.44 berechnet sich die Position der beats aus

$$\gamma_{i,b}^* = (t_i^* - \alpha_C^*) + b\tau_C \tag{3.62}$$

mit der Einschränkung $\gamma_{i,B}^* < t_i^* + B_h$.

3.2 Spektrale Merkmale

3.2.1 MFCC

MFCC's (*Mel Frequency Cepstral Components*) werden vor allem in der Sprachanalyse eingesetzt und repräsentieren das Spekrum der Einhüllenden eines logarithmierten, geglätteten Kurzzeitspektrums (mittels *Short time Fourrier Transformation, STFT*), d.h sie beschreiben weitgehend die Klangfärbung aufgrund der

spektralen Zusammensetzung. Ausgangspunkt ist dabei die gehörangepasste Tonhöhenwahrnehmung, die durch die Mel-Frequenzskala modelliert wird. Die lineare, äquidistante Frequenzauflösung der $STFT$ wird dabei auf die logarithmische Mel-Frequenzauflösung nach folgender Beziehung (vgl. Gleichung 3.63) umgerechnet. Die resultierende Energieverteilung kann dann in Folge in dreiecksförmig gewichtete Bänder gleicher Bandbreite zusammengefasst werden. Die geglättete Energieverteilung wird anschließend noch logarithmiert und in den Cepstralbereich transformiert [17].

$$f_{mel}(f) = 2595 * lg(1 + \frac{f}{700Hz}) \tag{3.63}$$

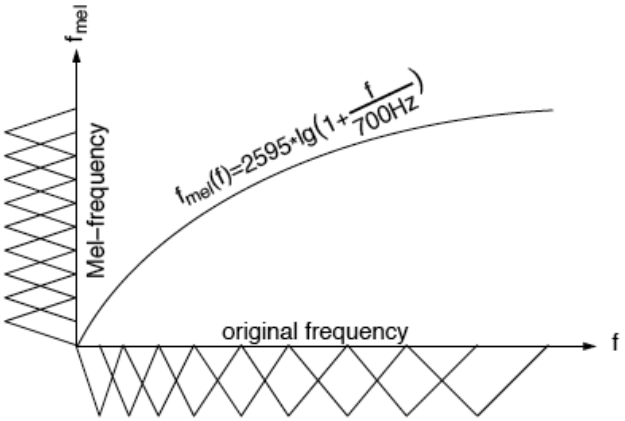

Abbildung 3.15: Mel - Frequenz mit Filterbänken

Die Berechnung der MFCC's pro frame erfolgt prinzipiell in 5 Schritten:

1. Betragsspektrum Berechnen ($STFT$)
2. Verteilung auf Mel-Skala abbilden
3. Energie innerhalb der kirtischen Bänder (CB) aufsummieren und gewichten
4. Gewichtete Energien logarithmieren
5. Transformation in den Cepstralbereich, z.B. mit Hilfe er dikreten Kosinus Transformation (DCT)

In der Mel-Skala werden zusammengehörige Frequenzen verwendet, um eine Komponente der MFCC's zu berechnen. Zusammengehörende Frequenzen sind jene Frequenzen, die innerhalb des gleichen kritischen Bandes (CB) liegen.

Um die Verwendung solcher Filterbänke und damit Probleme mit Diskretisierung und Filterform zu vermeiden, können MFCC's direkt aus dem Leistungsspektrum berechnet werden, indem die Abbildung auf die Mel-Skala direkt in der DCT eingerechnet wird [17]. Allgemein können Cepstral-Koeffizienten aus folgender Gleichung berechnet werden:

$$c_k = \frac{1}{2\pi} \int_{-\pi}^{\pi} lg \mid X(e^{j\omega}) \mid e^{j\omega k} d\omega. \tag{3.64}$$

Führt man nun eine Abbildungsvorschrift $\omega \rightarrow \tilde{\omega} = g(\omega)$ ein, ergibt sich obiges Integral zu

$$c_k = \frac{1}{2\pi} \int_{-\pi}^{\pi} lg \mid X(e^{jg^{-1}(\tilde{\omega})}) \mid e^{j\tilde{\omega}k} d\tilde{\omega}. \tag{3.65}$$

Ändert man die Integrationskonstante auf $d\tilde{\omega}/d\omega$ und approximiert das Integral durch eine Summe, so erhält folgende Gleichung:

$$c_k = \frac{1}{N} \sum_{n=0}^{\frac{N}{2}-1} \left\{ lg \mid X(e^{j\frac{2\pi n}{N}}) \mid cos \left[g\left(\frac{2\pi n}{N}\right) k \right] g'\left(\frac{2\pi n}{N}\right) \right\}. \tag{3.66}$$

Als Abbildungsvorschrift wird die Umrechnungsformel der Mel - Skala verwendet, wobei noch auf πnormalisiert werden muss:

$$g(\omega) = d \cdot lg\left(1 + \frac{\omega f_s}{2\pi \cdot 700Hz}\right) \tag{3.67}$$

mit

$$d = \frac{\pi}{lg\left(1 + \frac{f_s}{2 \cdot 700Hz}\right)}$$

3.2.2 Constant - Q - Tranformation

Eine weitere Möglichkeit für die Beschreibung der spektralen Eigenschaften eines Musikstückes stellen die Koeffizienten, die durch eine konstante Güte-Transformation (CQ-Transformation) bestimmt werden dar. Die konstant-Q-Transformation wurde in [6] eingeführt und ist der Fourriertransformation sehr ähnlich. Sie besteht aus einer Bank von Filtern, jedoch mit dem Unterschied, dass die *center*-Frequenzen der Filter geometrisch angeordnet sind $f_k = f_0 2^{\frac{k}{b}}$ $(k = 0, \ldots)$, wobei b die Anzahl der Filter pro Oktave angibt, sprich die Auflösung. Wählt man die Bandbreite des k-ten Filters mit $\Delta_k^{cq} = f_{k+1} - f_k = (2^{\frac{1}{b}} - 1)$ ergibt sich ein konstantes Verhältnis zwischen Frequenz und Auflösung:

$$Q = \frac{f_k}{\Delta_k^{cq}} = \frac{1}{2^{\frac{1}{b}} - 1}. \tag{3.68}$$

Wähl man nun $b = 12$ so erhält man, ausgehend von einer Grundfrequenz f_0 eine Unterteilung des Spektrums in Halbtöne und somit eine sehr gute musikalische Beschreibung. Weiters ergibt sich durch die Konstant-Q-Transformation eine bessere zeitliche Auflösung zu den höheren Frequenzen hin, was das menschliche Gehör nachbildet.

Die Filterkoeffizienten werden ausgehend von der Formel für das Fourier- Transformationsfilter an einer Frequenz z

$$\sum_{n<N} x[n] e^{-j2\pi nz/N} \tag{3.69}$$

abgeleitet. Die einzelnen Komponenten der Konstant-Q-Transformation (kq-bins) werden anhand solcher Filter berechnet, wobei die jeweils richtigen Werte für z und Fensterlänge N gefunden werden müssen. Die Bandbreite in 3.69 ist konstant $\Delta_z^{ft} = f_s/N$ und von z unabhängig. Die gewünschte Bandbreite $\Delta_k^{cq} = f_k/Q$ kann deshalb durch Wählen einer geeigneten Fensterlänge $N_k = \frac{f_s}{\Delta_k^{cq}} = Q\frac{f_s}{f_k}$ realisiert werden. Das Verhältnis Frequenz-zu-Auflösung in 3.69 ist $\frac{f_z}{\Delta_z^{ft}} = z$. Um nun einen konstanten Wert für diese Verhältnis zu erhalten setzt man $z := Q$. Damit ist für

ganzzahlige Werte Q das k-te bin der Konstant-Q-Transformation das Q-te DFT-bin mit der Fensterlänge $Q\frac{f_s}{f_k}$.

Zusammenfassend kann man die Transformation folgendermaßen berechnen: Zuerst wird eine Minimalfrequenz f_0 und eine Filteranzahl pro Oktave b gewählt. Um die Anzahl der zu berechnenden Koeffizienten einzuschränken wird ein Maximalfrequenz f_{max} als Einschränkung angegeben. Für die Transformation sind nun folgende Berechnungen notwendig [4]:

$$K := \lceil b \cdot \log_2(\frac{f_{max}}{f_0}) \rceil \quad \ldots \text{Anzahl der zu Koeffizienten} \tag{3.70}$$

Der $\lceil, ..., \rceil$- operator bedeutet hier, dass auf die nächste ganze Zahl aufgerundet wird.

$$Q := \frac{1}{2^{\frac{1}{b}} - 1} \tag{3.71}$$

$$N_k := \lceil Q\frac{f_s}{f_k} \rceil \tag{3.72}$$

$$x^{cq}[k] := \frac{1}{N_k} \sum_{n<N_k} x[n] w_{N_k}[n] e^{-j2\pi nQ/Nk} \tag{3.73}$$

wobei für $w_N[n]$ ein Hanning-Fenster verwendet wird [6, 7].

3.3 Musikalische Merkmale

3.3.1 Chroma- Feature

In der Chroma-Darstellung wird ein Ton durch seine Tonhöhe h und seine Pitch-Klasse c bestimmt. Die Tonhöhe steht dabei für die Oktavlage des Tones, die Tonklasse gibt seine chromatische Zugehörigkeit an (zum Beispiel $A^\sharp$) an. Somit kann kann die wahrgenommene Tonhöhe p (pitch) wie folg dargestellt werden:

$$p = 2^{h+c} \tag{3.74}$$

Somit ergibt sich eine zyklische Darstellung in Form eines Helix. Erhöht sich die Tonhöhe von $C1$ nach $C2$, so werden alle Pitch-Klassen auf dem Helix durchlaufen.

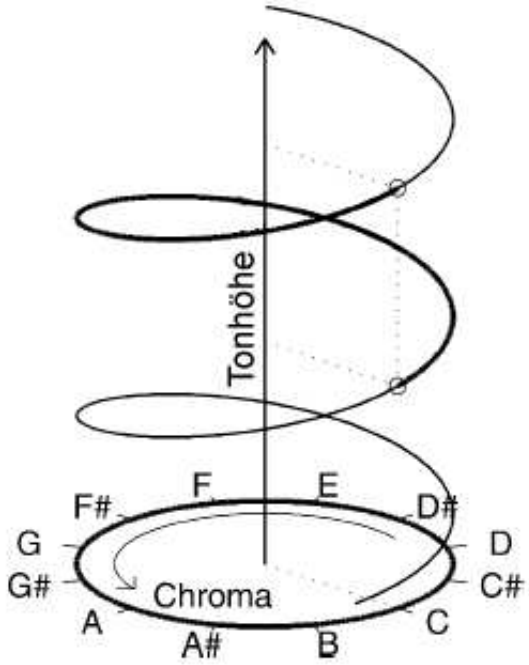

Abbildung 3.16: Helix zur Chroma-Beschreibung. Die vertikale Achse gibt die Tonhöhe an, der Winkel enthält die Pitch-Klassen-Information (vgl. [3])

Es ergibt sich eine hinreichende Beschreibung für alle Töne, wenn $c \epsilon [0, 1)$ und $h \epsilon \mathbb{Z}$. [3]

Teilt man das Intervall zwischen 0 und 1 in 12 gleiche Teile ein, so erhält man die 12 Halbtöne der chromatischen Skala. Teilt man für jedes Frame nun die vorkommenden Frequenzen auf die Pitch-Klassen auf, indem man diese über alle Oktaven aufsummiert, so erhält man einen *Feature*-Vektor für jedes Frame.

Der Chromawert kann aus der Frequenz wie folgt berechnet werden:

$$c = log_2 f - \llcorner log_2 f \lrcorner \tag{3.75}$$

wobei $\llcorner \cdot \cdot \cdot \lrcorner$ für die größte ganze Zahl steht. Das heißt, Chroma ist der fraktionelle Teil des 2er Logarithmus. Dieser Wert für c wird nun über das Spektrum berechnet,

wobei dieses auf einen Bezugston (27.5Hz) normiert wird. Nun wird jeder Bereich im Spektrum auf seine zugehörige Pitch-Klasse abgebildet. Für fedes Frame werden die Frequenzen, welche zur gleichen Klasse gehören aufsummiert. So ergibt sich für jedes Frame ein 12-dimensionaler Chromaektor und somit die Chromainformation über die Zeit.

$$v_{t,k} = \sum_{n \epsilon S_k} \frac{F_t(n)}{N_k}, \quad k \, \epsilon \{0...11\} \tag{3.76}$$

Kapitel 4

Berechnung der Ähnlichkeit

Aus den erhaltenen Eigenschaften wird nun versucht, auf die Struktur des Stückes zu schließen, beziehungsweise Wiederholungen zu finden. Gefundene Wiederholungen werden dann auf Grund ihrer Ähnlichkeit als Thumbnails verwendet oder nicht.

4.1 Korrelation

Ausgehend von einer Matrix, welche die beschreibenden *features* enthält, wird eine Korrelationsmatrix berechnet. Dazu wird jeder *feature*-Vektor mit einem um τ verschobenen Vektor verglichen, wobei τ so gewählt wird, dass jeder Vektor mit jedem verglichen wird.

$$\rho(\tau) = \int_{-\infty}^{+\infty} x(t)x(t+\tau)dt. \tag{4.1}$$

Durch die Korrelation erhält man eine Matrix, welche in der Hauptdiagonale durchwegs den Wert 1 aufweist und in der man anhand der Nebendiagonalen auf Wiederholungen schließen kann. Je ausgeprägter eine Nebendiagonale ist, desto größer ist die Ähnlichkeit der Teile. Abbildung 4.1 zeigt die Korrelationsmatrix des Stückes *Margaretaville* von Jimmy Buffet aus [3].

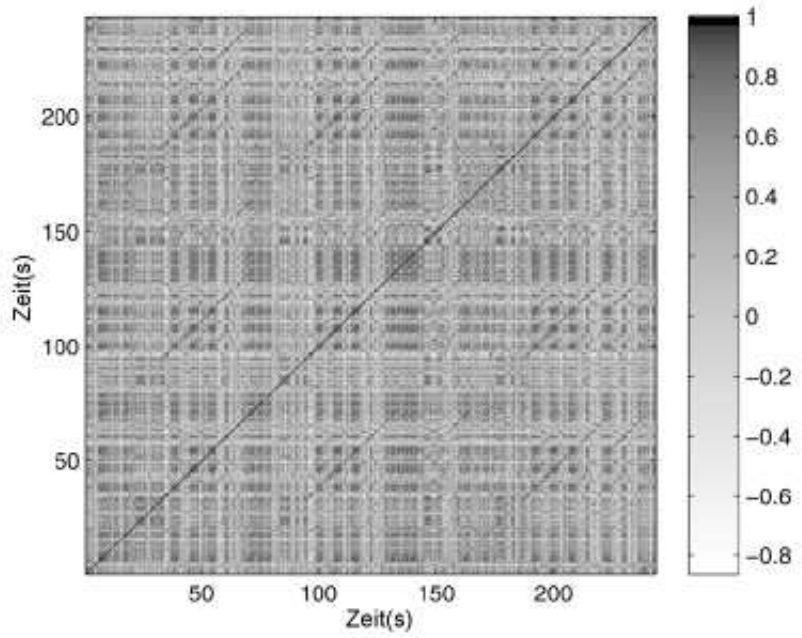

Abbildung 4.1: Beispiel einer Autokorrelationsmatrix

4.2 Hough Transformation

Die Hough-Transformation ist ein Werkzeug zur Detektion von Linien in einem Bild. Eben eine solche Detektion von Linien wird in [2] verwendet, um Diagonalelemente in einer Matrix zu finden. Da nicht nur 45° Diagonalen vorkommen können (zum Beispiel aufgrund von Temposchwankungen) ist es notwendig, auch Diagonalen mit etwas größeren beziehungsweise kleineren Winkeln in Betracht zu ziehen.

Die Transformation basiert auf einer Abbildung in den Parameterraum (m, c-Raum). Alle durch ein Pixel (x, y) in einem Bild laufenden Gerade folgen der Gleichung

$$y = mx + c \quad \forall (m, c) \in \Re^2. \tag{4.2}$$

Eine gegebene Linie, welche durch (x, y) läuft kann vollständig durch die Koordinate im Parameterraum (m_0, c_0) beschrieben werden. Im Parameterraum entsprechen

alle Linien durch den Punkt (x, y) allen Punkten (m, c) für welche

$$c = -mx + y \tag{4.3}$$

gilt. Dies ergibt im Parameterraum wiederum ein Set von Geraden.

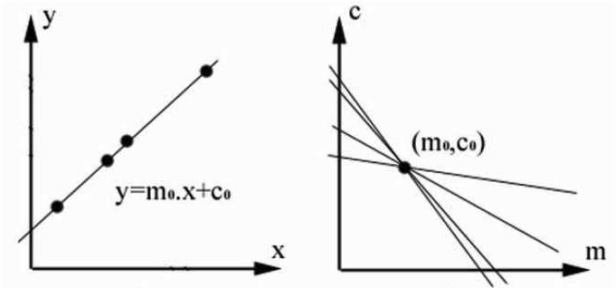

Abbildung 4.2: Gerade und Abbildung im Parameterraum (siehe [2])

Der Schnittpunkt (m_0, c_0) der Geraden in Abbildung 4.2 ergibt die Parameter für die gesuchte Gerade. Dieses Mapping wird für alle Pixel im Bild wiederholt, um einen Parameterraum zu erhalten. In diesem kann man nun durch finden der Maxima die Steigung m und Verschiebung c der besten Geraden auslesen. Jedes lokale Maximum (m_0, c_0) entspricht einer Geraden im Bild, sprich in der Korrelationsmatrix. Die zwei Pfeile in Bild 4.3 zeigen die Position der im linken Bild eingezeichneten Diagonalen im Parameterraum an.

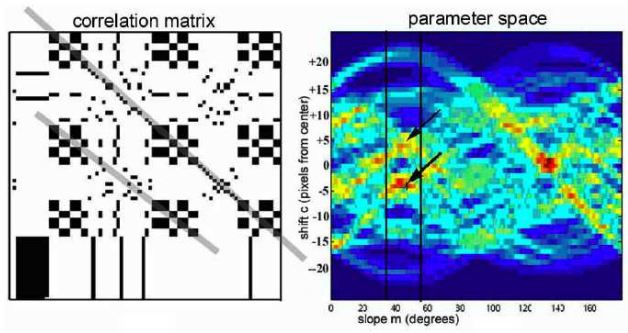

Abbildung 4.3: Korrelationsmatrix mit zugehörendem Parameterraum (vgl. [2])

Der große Vorteil dieser Transformation ist, dass nicht nur die 45° Geraden, son-

dern alle Geraden gefunden werden. Um die Suche einzugränzen, werden nur Winkel zwischen ungefähr 40° bis 50° bei der Berechnung berücksichtigt. Entlang der erhaltenen Geraden werden jetzt die größten Ähnlichkeiten in der Korrelationsmatrix gesucht, indem ein Schwellwert eingeführt und die entsprechenden Pixel auf eins, alle anderen auf null gesetzt werden. Somit ergibt sich eine Matrix, in der nur entlang der Diagonalen Werte (1) stehen. Benutzt man diese Matrix als Maske für die Korrelationsmatrix, sieht man nurmehr die Diagonalen mit dem größtem Korrelationsgrad und dadurch die Wiederholungen.

4.3 Filterung der Matrix

Zum Finden von Wiederholungen wird die Ähnlichkeit zwischen Segmenten im Musikstück mit konstanter Zeitdifferenz berechnet. Dazu wird entlang der Diagonalen der Ähnlichkeitsmatrix gefiltert und die so erhaltene Matrix wird so gedreht, dass die Diagonalen vertikal orientiert sind. Die Berechnung dazu sieht folgendermaßen aus:

$$T_{i,j} = \sum_k C_{i+k,i+j+k} w(k). \tag{4.4}$$

$w(k)$ ist dabei die Impulsantwort des Filters, welches hier ein *moving average*-Filter ist. T ist die gefilterte Matrix und C die Korrelationsmatrix beziehungsweise die Diagonalenmatrix.

Das (i, j)-te Element von T steht für die Ähnlichkeit zwischen dem Segment mit Startzeitpunkt i und jenem mit Startzeitpunkt $i+j$. Das bedeuted i ist der Zeitindex und j steht für die Verzögerung zwischen den Segmenten.

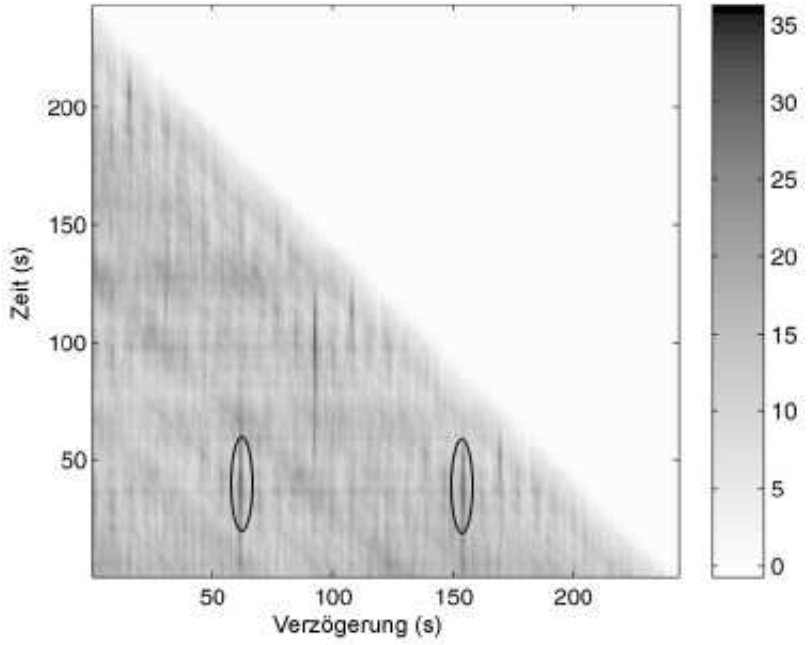

Abbildung 4.4: Gefilterte Matrix der obigen Korrelationsmatrix (vgl. [3])

Die Auswahl des *fingerprints* geschieht nun durch finden von Maxima in der gefilterten Matrix. Entscheidend hierfür ist neben der Position des Maximums auch die Länge des Fensters $w(k)$, welche gleichzeitig die Länge für den Fingerabdruckes darstellt. Das bedeutet, dass die Fensterlänge sehr geschickt gewählt werden muss und nicht für alle Titel und Genres generelle Gültigkeit hat. In Abbildung 4.4 ergäbe dies eine Wiederholung des Teiles, welcher bei circa 42 Sekunden beginnt mit einer Verzögerung von circa 62 Sekunden und nochmal nach ungefähr 155 Sekunden (siehe markierte Stellen in Abbildung 4.4). Das bedeutet, der 42 Sekunden im Stück beginnende Teil wiederholt sich 62 Sekunden später. Die Auswahl des *thumbnails* ist also von 42 Sekunden bis 42 Sekunden Plus der Fensterlänge (im Beispiel 30 Sekunden [3])

Kapitel 5

Test der einzelnen Verfahren

Um die verschiedenen Ansätze der einzelnen Kapitel zu testen, wurde eine Datenbank aus 8 Musikstücken herangezogen. Davon fallen fünf ins Genre Pop, die restlichen sind aus verschiedenen Richtungen (Swing, Marsch und Klassik). Das Hauptaugenmerk liegt dabei bei Pop-Musik, da die meisten der vorgestellten Methoden eben für dieses Genre ausgelegt wurden. Die Signale wurden dabei jeweils als Monosignale eingelesen, mit einer Samplingfrequenz von 44,1kHz und einer Auflösung von 16 Bit.

Zur Untersuchung der verschiedenen Verfahren wurden folgende Musikstücke gewählt:

	Titel	***Interpret***	***Länge***
Pop1	*Flake*	*Jack Johnson*	*4'58"*
Pop2	*Slow Down Baby*	*Christina Aguilera*	*3'29"*
Pop3	*Startting Today*	*Natalie Imbruglia*	*2'54"*
Pop4	*Stop Loving You*	*Toto*	*4'30"*
Pop5	*With Or Without You*	*U2*	*4'58"*
Swing	*I've Got You Under My Skin*	*Frank Sinatra*	*3'43"*
Marsch	*Radetzky Marsch*	*Johann Strauß*	*2'39*
Klassik	*Serenade in C Op. 48 II: Waltz*	*Pjotr Iljitsch Tschaikowski*	*3'52"*

Tabelle 5.1: Auflistung der verschiedenen Testsongs

Die fünf Pop-Titel sind so gewählt, dass sie verschiedene Eigenschaften aufweisen. *Pop1* ist ein Akustikgitarren-Stück mit Schlagzeug und Gesang, das sehr perkussi-

ve Eigenschaften hat. *Pop2* geht eher in die Richtung R'n'B[1], hat daher sehr viel Energie in den tiefen Frequenzen und eine etwas schwierigere rhythmische Struktur. *Pop3* ist ein typischer Pop-Song[2] mit einer dem Genre typischen Struktur. *Pop4* hat eine aufwendigere Instrumentalisierung als die anderen, weist aber ansonsten eine ähnliche Struktur. *Pop5* weist auch eine einfache[3] Struktur auf und eine genauso einfache Instrumentalisierung. Bei allen Titeln aus dem Genre Pop ist die Struktur ähnlich, d.h das Musikstück besteht aus zwei bis drei Wiederholung des Refrain und mehreren Strophen. Bei einigen Stücken wiederholen sich auch Microstrukturen mehrmals im Stück. Für den *Swing* gilt ähnliches wie für die Pop-Songs, jedoch mit völlig anderen Instrumenten (Big-Band) und einer anderen Rhythmussrtuktur. Der *Marsch* sollte sich deshalb gut eignen, da die *onsets* beziehungsweise der *beat* sehr klar markiert sind und es sozusagen eins-zu-eins Wiederholungen der Teile gibt. Im klassischen Stück wiederholt sich das Hauptthema mehrmals in verschiedenen Registern (tiefen und hohen Streichern), die Übergänge sind eher gleitend und nicht perkussiv.

5.1 Test der Onset - Detection

Für eine statistische Auswertung der verschiedenen *onset*-Detektionsverfahren wurden von jedem Song nur die jeweils ersten 20 Sekunden verwendet, da die einzelnen Verfahren konsistent innerhalb eines Stückes sind. Dies wurde Anhand eines Stückes, das für alle Verfahren vollständig getestet wurde untersucht. Dabei wurden mittels des *Sound Onset Labelizers*[4] von Pierre Leveau [16] die *onsets* manuell markiert.

[1] *Rhythm and blues*

[2] Ein typischer Pop-Song beginnt mit einer kurzen Einleitung (*Intro*) es folgt meistens die erste und zweite Strophe, dann das erste Mal der Refrain. Nach diesem kommt eine weitere Strophe, dann nochmal der Refrain (eventuell in variierter Form) oder ein Zwischenteil (*Bridge*). Am Schluss steht meistens nochmal der Refrain, sehr oft mit einem *fadeout*.

[3] *Intro, Strophe, Strophe, Refrain, Strophe, Refrain, Schluss (AABAB)*

[4] Der *Sound Onset Labelizer* und eine dazugehörige Beschreibung stehen unter www.lam.jussieu.fr/src/Membres/Leveau/SOL/SOL.htm zum Download zur Verfügung

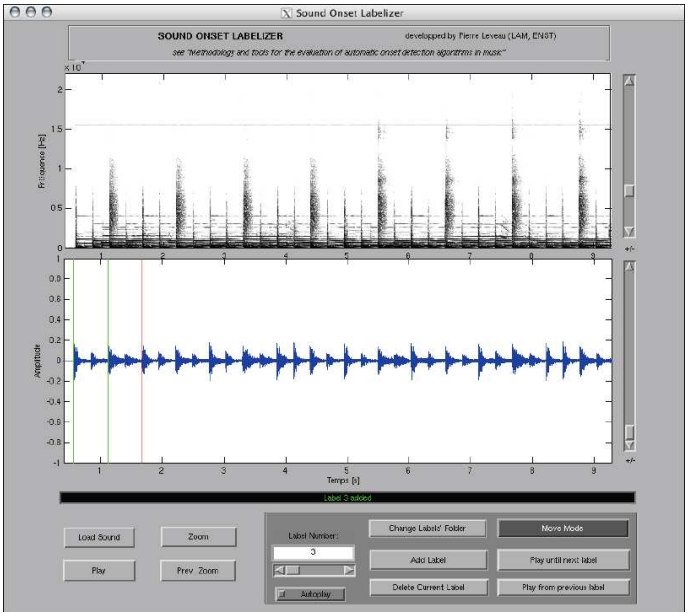

Abbildung 5.1: *Sound Onset Labelizer* von Pierre Leveau

Die untere Hälfte in Abbildung 5.1 zeigt die Amplitude des Musiksignals, (*U2 - With or Without You*), die obere das Spektrogramm. Nachdem für alle Stücke die ersten 20 Sekunden gelabelt wurden, wurde diese Onsetfunktion als Referenz für die verschiedenen Verfahren herangezogen.

Für die statistische Auswertung wurden drei Parameter berechnet:

1. Die Anzahl der richtig detektierten Onsets (*TP, total positive*)
2. Die Anzahl der falsch detektierten Onsets (*FP, false positive*)
3. Die Anzahl der nicht detektierten Onsets (*FN, false negative*)

Diese wurden in Relation zur gesamten *onset*-Anzahl (*TO, total onsets*) aus dem *labelizer* gesetzt um Werte in Prozent zu erhalten.

$$TP[\%] = \frac{TP}{TO} * 100 \tag{5.1}$$

$$FP[\%] = \frac{FP}{TD} * 100 \tag{5.2}$$

$$FN[\%] = \frac{FN}{TO} * 100 \tag{5.3}$$

Bei der Berechnung der *false positives* wurde als Referenzwert die Anzahl der total detektierten (*TD, total detected*) onsets herangezogen. Solche Werte werden auch in der Literatur durchwegs verwendet (vgl. [10,12]) um Aussagen über die *performance* zu machen. In [12] werden nur TP und FP verwendet und gegeneinand geplottet, Dixon [10] verwendet noch zwei weitere Werte (*precision* und *recall*), die sich aus den oben genannten Werten berechnen. Für eine klare Aussage zur Leistung der Verfahren reichen jedoch die in Prozent berechneten Werte, da eine sehr gute Veranschaulichung bzw. Darstellung der erzielten Ergebnisse geben.

Wegen der Ungenauigkeiten beim Markieren der Onsets von Hand wurde für alle drei Werte (*TP, FP, FN*) eine Toleranz von 25ms[5] (vgl. [10,12]) zugelassen, d.h.: wenn ein Onset innerhalb eines 50ms Fensters vom wahren Zeitpunkt detektiert wurde, so wurde ein richtig detektierter Onset (TP) gezählt, andernfalls ein falsch detektierter (FP). (vgl. Abbildung 5.1)

Das Abzählen der Onsets beziehungsweise Zuordnen der Onsets wurde nach dem *peak-picking* gemacht. Die Parameter λ und M für das *peak-picking* wurden dabei für alle Stücke gleich belassen ($\lambda = 0.9$, $M = 100ms$), während δ manuell an die jeweilige *onset*-Funktion angepasst werden musste, um die Leistung beziehungsweise Präzision zu steigern[6].

Als Maß für die Performance des jeweiligen Verfahrens wurde das Verhältnis zwischen richtig detektierten (TP) und der Summe aus falsch detektierten (FP) und nicht detektierten (FN) $\frac{TP}{FP+FN}$ verwendet.

Folgende Verfahren wurden getestet:

- *prob*: Heuretisches Verfahren (Kapitel2.6)
- *lpc*: Linear Prediction Coding (Kapitel 2.8)

[5] 25ms entspricht in etwa einer Sechzehntelnote bei 600bpm

[6] Aufgrund des hohen Berechnungsaufwandes des *peakpicking* bzw. des Medianfilters wurde jedoch δ nur einmal angepasst, und nicht wie in der Literatur mehrere Male.

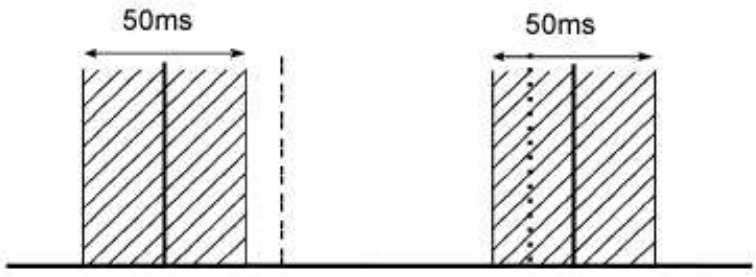

Abbildung 5.2: Toleranzfenster für die *onset*- bzw. die *beat-detection*. Die durchgezogene Linie zeigt die wahre Position, die strichlierte eine falsch positiv markierte und die gepunktete eine richtig positiv markierte Position. Im linken Fenster würde eine fälschlich negativ markierte Position gezählt.

- *NWPD*: Normalized Weighted Phase Deviation (Kapitel 2.4)
- *WPD*: Weighted Phase Deviation (Kapitel 2.4)
- *PD*: Phase Deviation (Kapitel 2.4)
- *Hyb 3*: Hybrider Ansatz mit zeitlich basierter Methode (Kapitel 2.2) in tiefen Frequenzen und euklidischem Abstand (Kapitel 2.5) im höchsten Band
- *Hyb 2*: Hybrider Ansatz, spektrale Domäne (Kapitel 2.5) in allen Bändern
- *Hyb 1*: Hybrider Ansatz, spektrale Domäne im höchsten, Energiebetrachtung (Kapitel 2.2) in den tiefen Bändern
- *SD*: Spectral Difference (Kapitel 2.3)
- *Complex*: *onset-detection* in der spektralen Domäne (2.5)
- *time*: Zeitlich basierter Ansatz (Kapitel 2.2)

5.2 Test der Beat - Detection

Bei der *beat-detection* mittels *onset-detection* (Kapitel 3.1.2) wurde in gleicher Weise verfahren wie bei der *onset-detection*. Wiederum wurden mit Hilfe des *Onset-Labelizers* von [16] manuell jene Punkte markiert, an welchen der *beat* vorkommt. Bei der Auswertung der Ergebnisse lag der Schwerpunkt nicht auf dem detektierten Tempo, sondern auf der Position der markierten Punkte, da nur diese im Rahmen dieser Arbeit Relevanz haben. Dies deshalb, weil eine Wiederholung von Rhythmusmustern keine Information zu Wiederholungen einbrachte, bzw nicht zur Suche nach Wiederholungen verwendet wurde.

Auch bei der *beat-detection* wurden wieder nur die ersten 20 Sekunden des jeweiligen Stückes verwendet. Dies ist zwar nicht ganz korrekt, da die *beat-detection* mittels *onsets* nicht immer konsistent im ganzen Stück ist, doch für eine qualitative Aussage soll hier vorerst ein kurzer Aussschnitt reichen. Ähnlich der *onset-detection* wurden die von Hand markierten Positionen der *beats* als die die Gesamtanzahl der vorkommenden beats BT für den Referenzwert herangezogen. Die Anzahl der vom Algorithmus detektierten *beats* wurde mit BD (*beats detektiert*) bezeichnet. Nun wurden richtig (*BR, beats richtig*) und falsch (*BF, beats falsch*) detekierte Positionen gezählt und in Prozent umgerechnet:

$$BR[\%] = \frac{BR}{BD} * 100 \tag{5.4}$$

$$BF[\%] = \frac{BF}{BD} * 100 \tag{5.5}$$

Da bei den falsch detektierten *beat*-Positionen auch diejenigen mitgezählt wurden, welche auf dem *off-beat* liegen, wurden diese nochmals separat gezählt und mit OB (*off - beat*) angegeben, da diese im musikalischen Sinne nicht zwingend falsch sein müssen, beziehungsweise da auch das doppelte oder halbe Tempo zulässig ist. Auch hier wurde eine etwas größere Toleranz zugelassen (30ms).

Auch das Tempo in BPM (*beats per minute*) wurde berechnet, allergdins nur aus dem durchschnittlichen Abstand $\overline{\Delta}$ der Positionen, was zu einem relativ großen

Fehler führt, wenn der Algorithmus mehr Positionen findet als tatsächlich vorhanden sind. Dazu wurde ein Histogram der Distanzen erstellt und daraus das Tempo berechnet:

i	Anbstand Δ[s]	Häufigkeit N
1	0.5820	1
2	0.5950	2
3	0.6079	6
4	0.6209	8
5	0.6338	1
6	0.6468	2
7	0.6597	8
8	0.6727	0
9	0.6856	1
10	0.6986	1
		$\sum$31

Tabelle 5.2: Histogram der von Hand markierten *beat* - Positionen am Beispiel *Radektzky - Marsch*

Das Tempo wurde dann wie folgt berechnet:

$$\overline{\Delta} = \frac{1}{N_t} \sum_{i=1}^{k} \Delta_i N_i \tag{5.6}$$

$$BPM = 60/\overline{\Delta} \tag{5.7}$$

N_t ist die gesamte Anzahl der Abstände. Diese Berechnung ist jedoch nur sinnvoll, bzw. realistisch, wenn die Anzahl der detektierten *beat-locations* mit der tatsächlichen Anzahl dieser übereinstimmt.

5.3 Test der Feature extraction

Bei der *feature-extraction* wurde in zwei Richtungen getestet: Zum einen wurden als Referenzpunkte für die *frames* die gefundenen *onsets* verwendet, zum anderen die *beat*-Positionen. Die *features* waren dabei drei Verschiedene *chroma*-Vektoren,

dreimal jeweils mittels verschiedener Verfahren extrahierte *MFCC*'s und ein Set an konstant-Q-Koeffizeinten.

In der folgenden Tabelle sind kurz die Unterschiede in den Methoden erklärt:

Kürzel	Beschreibung
chroma1	chroma Berechnung über die ganze Bandbreite (0Hz bis 22kHz)
chroma2	Banbegrenzung auf 20 bis 200Hz
chroma3	Aufteilung in 3 Bänder (63,5 bis 240 Hz, 240 bis 1020Hz und 1020 bis 4070Hz)
MFCC1	Berechnung der MFCC's direkt aus dem Leistungsspektrum [17]
MFCC2	MFFC's aus diskreten Dreiecksfiltern
MFCC3	Programm von Dan Ellis [7]
cq-coeffs	48 Konstant-Q-Koeffitienten pro Oktave von 20Hz bis 4kHz [8]

Tabelle 5.3: Kurze Beschreibung der verwendeten *feature-extraction* Verfahren

Bei den MFCC's wurden jeweils 36 Koeffitienten berechnet. Die Verwendeten *frames* waren dabei immer um einige Millisekunden kleiner als die eigentlichen Abstände zwischen zwei markierten Punkten, um hochfrequente Anteile im *attack* der Noten weitgehend auszublenden.

5.4 Finden von Ähnlichkeiten

Die pro *frame* errechneten *feature*-Vektoren wurden jeweils in eine Matrix eingetragen und daraus für jede dieser Matrizen eine *Autokorrelation* (Kapitel 4.1) durchgeführt. Für jede dieser Autokorrelationsmatrizen wurde die *Hough-Transformation* aus Kapitel 4.2 und die *Filterung der Matrix*, wie sie in Kapitel 4.3 beschrieben ist, realisiert. Da die Houghtransformation in fast allen fällen keine brauchbaren Ergebnisse lieferte, wurden die Ähnlichkeiten nur mittels der gefiterten Matrix gesucht. Dies geschah durch Suchen von Maxima in der Matrix.

Die Ergebnisse dieser Ähnlichkeitsfindung wurden durch Anhören der gefundenen *Thumbnail*-Kandidaten überprüft. Die Fensterlänge für die Matrizenfilterung wurde dabei experimentell Angepasst. Da die Matrix nicht über konkrete Zeitpunkte, sondern über eine bestimmte Anzahl von *frames* gefiltert wurde, mussten die Zeitpunkte der frames aus den *onset*- bzw. *beat*-Positionen berechnet werden. Dabei ist zu

sagen, dass nicht immer gleichviele *onset* oder *beats* im gleichen Zeitraum vorhanden sein müssen, weshalb eine konstante Fensterlänge eigentlich nicht für das ganze Stück Gültigkeit hat.

Kapitel 6

Ergebnisse und Vergleich der verschiedenen Ansätze

6.1 Ergebnisse der Onset Detection

Methode	***TD***	***TP***	***FP***	***FN***	$\frac{TP}{FP+FN}$
prob	*691*	*173 (24%)*	*518 (75%)*	*554 (76%)*	*0.16*
lpc	*713*	*269 (37%)*	*444 (62%)*	*458 (63%)*	*0.30*
NWPD	*839*	*330 (45%)*	*509 (61%)*	*397 (55%)*	*0.36*
WPD	*449*	*226 (31%)*	*223 (50%)*	*501 (69%)*	*0.31*
PD	*753*	*319 (44%)*	*434 (58%)*	*408 (56%)*	*0.38*
Hyb 3	*834*	*227 (31%)*	*607 (73%)*	*500 (69%)*	*0.21*
Hyb 2	*129*	*44 (6%)*	*85 (66%)*	*683 (94%)*	*0.06*
Hyb 1	*115*	*20 (3%)*	*95 (83%)*	*707 (97%)*	*0.02*
SD	***393***	***256 (35%)***	***137 (35%)***	***471 (65%)***	***0.42***
Complex	*379*	*200 (28%)*	*179 (47%)*	*527 (72%)*	*0.28*
HFC	*523*	*196 (27%)*	*327 (63%)*	*531 (73%)*	*0.23*
time	*398*	*109 (15)*	*289 (73%)*	*618 (85%)*	*0.12*
# von Hand markierte Onsets:				*727*	

Tabelle 6.1: Tabelle der Ergebnisse der *onset-detection* Verfahren über alle Songs zusammen

Für die fünf Pop-Songs ergab die *spectral difference* (Kapitel 2.3) das beste Ergebnis mit 53% richtig detektierten Onsets im Mittel bei nur 20% *false positives* und

47% *false negatives*. Bei drei dieser Pop-Songs war auch diese Methode die beste Wahl (Pop3 [56%, 28%, 44%], Pop2 [52%, 17%, 48%] und Pop5 [46%, 10%, 54%] für [*TP*, *FP*, *FN*]). Vor allem die geringe Anzahl der fälschlicherweise positiv detektierten Onsets ist bei diesem Verfahren sehr gering und zieht deshalb den Wert für die Gesamtleistung nach oben. Für die zwei restlichen Pop-Songs konnten mittels *normalized weighted phase deviation* (*NWPD*, Gleichung 2.12 in Kapitel 2.4) und *phase deviation* (Gleichung 2.10 im Kapitel 2.4) die besten Ergebnisse erzielt werden (Pop1 [63%, 34%, 37%] und Pop4 [69%, 17%, 31%]). Untenstehende Tabelle (Tabelle 6.1) zeigt die Ergebnisse aller getesteten Verfahren für das Genre Pop:

Methode	***TD***	***TP***	***FP***	***FN***	$\frac{TP}{FP+FN}$
prob	*355*	*120 (23%)*	*235(66%)*	*394 (77%)*	*0.19*
lpc	*463*	*253(49%)*	*210(45%)*	*261 (51%)*	*0.54*
NWPD	*514*	*266 (52%)*	*248 (28%)*	*248 (48%)*	*0.54*
WPD	*288*	*194(38%)*	*94(33%)*	*320 (62%)*	*0.47*
PD	*435*	*270(53%)*	*165 (38%)*	*244 (47%)*	*0.66*
Hyb 3	*448*	*139 (27%)*	*309(69%)*	*375 (73%)*	*0.20*
Hyb 2	*106*	*6(1%)*	*100(94%)*	*508 (99%)*	*0.01*
Hyb 1	*89*	*12 (12%)*	*77(87%)*	*502 (98%)*	*0.02*
SD	***338***	***270(53%)***	***68 (20%)***	***244 (47%)***	***0.87***
Complex	*289*	*171(33%)*	*118 (41%)*	*343(67%)*	*0.37*
HFC	*325*	*111(22%)*	*214(66%)*	*403 (78%)*	*0.18*
time	*253*	*91(15%)*	*162 (64%)*	*423 (82%)*	*0.16*
# von Hand markierte Onsets:				*514*	

Tabelle 6.2: Tabelle der Ergebnisse der *onset-detection* Verfahren für alle Pop-Songs

Erwartungsgemäß funktionierte die *onse-detection* für das klassische Stück am schlechtesten, da dort bereits das Markieren der Onsets von Hand große Schwierigkeiten bereitete. Dennoch konnte unter den verschiedenen Ansätzen ein Eindeutiger Favorit gefunden werden. Es war dies der LPC-Ansatz mit [20%, 82%, 80%]. Für den *Marsch* funktionierte der heuristische Ansatz am besten, die Perfomance lag bei [55%, 67%, 54%]. Beim Swing erzielte die Messung des *high-frequency-content* mit [44%, 15%, 56%] die besten Ergebnisse (siehe Tabelle 6.1).

Insgesamt ist zu sagen, dass die Ergebnisse im Vergleich zur Literatur sehr

schlecht ausfielen. Dies ist wohl darauf zurückzuführen, dass bei den Tests aus der Literatur die Systemparameter für das *peak-picking* nach der statistischen Auswertung in Hinsicht einer besten Perfomance optimiert wurden. Solch ein Verfahren konnte im Rahmen dieser Arbeit aufgrund des hohen zeitlichen Aufwandes des *peak-pickings* nicht durchgeführt werden.

Auch war die Kurvendiskussion, die zum Finden der Maxima in der *onset-detection* leider nicht ganz richtig, was die Performance der einzelnen Verfahren noch witr verschlechterte und somit die Erklärung für die im Vergleich zur Literatur so schlechten Werte ist. Leider wurde dies erst zu spät entdeckt, um es noch zu korrigieren. Die Ergebnisse sind im Anhang A im einzelnen aufgelistet.

6.2 Ergebnisse der Beat detection

Song	***BT***	***BD***	***BR***	***BF***	***OB***	***BPM*** *Algo.*	***BPM*** *Hand*
Flake	*32*	*37*	*24 (65%)*	*13 (35%)*	*3 (8%)*	*112*	*97*
Starting today	*30*	*31*	*23 (75%)*	*8 (25%)*	*3 (10%)*	*98*	*94*
Slow Down Baby	*32*	*33*	*29 (88%)*	*4 (12%)*	*0 (0%)*	*96*	*96*
Stop Loving You	*33*	*38*	*5 (13%)*	*33 (87%)*	*23 (70%)*	*118*	*100*
With or Without You	*36*	*36*	*36 (100%)*	*0 (0%)*	*0 (0%)*	*110*	*110*
Radetzky Marsch	*31*	*40*	*10 (25%)*	*30 (75%)*	*15 (37,5%)*	*124*	*95*
Waltz	*20*	*38*	*2 (5%)*	*36 (95%)*	*0 (0%)*	*114*	*60 [180]*
I've Got You Under My Skin	*42*	*42*	*36 (86%)*	*6 (14%)*	*0 (0%)*	*127*	*125*

Tabelle 6.3: Ergebnisse der *beat-detection* mittels *onset-detection*

Tabelle 6.3 zeigt die Ergebnisse der *bea-detection* für die jeweils ersten zwanzig Sekunden eines jeden Titels. Für die Pop-Songs ergab sich ein Durchschnitt von 68.2% richtig detektierter *beat* - Positionen. Dabei senkt sich sicherlich die Trefferrate durch die 70% *off-beat* detektierten Positionen im Song *Stop Loving You*. Beim Song *Slow Down Baby* wurde eine Position mehr am Ende der zwanzig Sekunden detektiert, welche zwar richtig ist, leider aber nicht mehr in den von Hand markierten Positionen aufscheint. Dies deshalb, weil es am Ende des Zeitfensters sehr schwierig war, die Position zu markieren. Also müsste der korrigierte Wert der richtig detektierten

Positionen mit 90% angegeben werden. Die 100% beim *U2*-Song sind damit zu erklären, dass es sich um ein sehr perkussives Stück handelt mit sehr gut definierter rhythmischer Struktur.

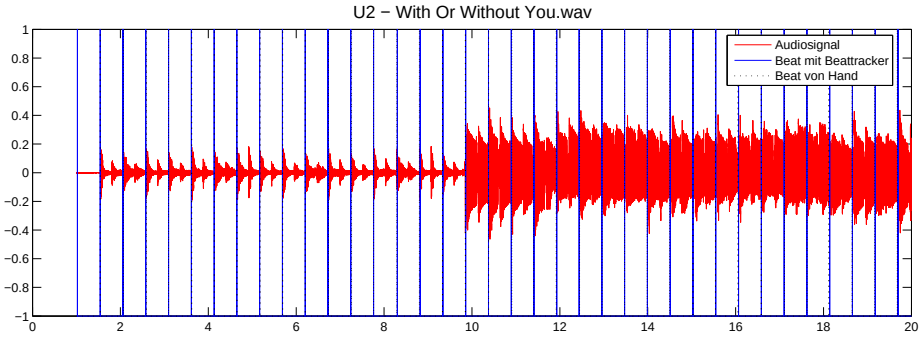

Abbildung 6.1: Beat detection für *U2 - With Or Without You*

Da am Beginn des Stückes fast eine Sekunde lang überhaupt kein Signal vorhanden ist und dann bis zum beginn etwa 0,4 Sekunden nur Rauschen, hat der Algorithmus bereits vor dem eigentlichen Anfang des Stückes eine Position für ein *beat*-Ereignis markiert. Dies geschah bei mehreren Musikstücken. Die falschen Markierungen wurden bei der statistischen Auswertung vernachlässigt, da diese falsch markierte Position vor Beginn des Stückes keine Auswirkungen auf die Funktionalität der Kodierung hat.

Eine sehr gute Performance ergab sich auch für den Swing von Frank Sinatra, bei dem mit 86% richtig detektierten Positionen ein guter Prozentsatz erreicht wurde. Beim Radetzky-Marsch wurde nur ein Viertel der Positionen gefunden. Zählt man diese mit den *off-beat* Positionen zusammen erhält man 62.5% an akzeptablen Werten. Für das klassische Stück hat der Algorithmus nicht funktioniert, vor allem auch deshalb, weil schon die *onset-detection* für diese Art von Signalen sehr schlecht funktioniert, da fast alle Noten gleitend beginnen, ähnlich einem Glissando. Der Wert der bei *BPM Hand* in eckigen Klammern steht kommt daher, dass bei der Markierung von Hand nur jeweils die “1”, also der Beginn jeden Taktes markiert wurde. Es ergäbe sich also für die Viertelnoten ein *beat* von $3 \times 60 = 180$. Weiters muss hier festgehalten werden, dass bereits das *hand-labeling* extrem schwierig war, da die

Startpunkte der einzelnen Noten sehr schwierig zu finden waren und sich in den einzelnen Instrumenten zeitlich überlagerten, beziehungweise nicht immer gleichzeitig vorkamen.

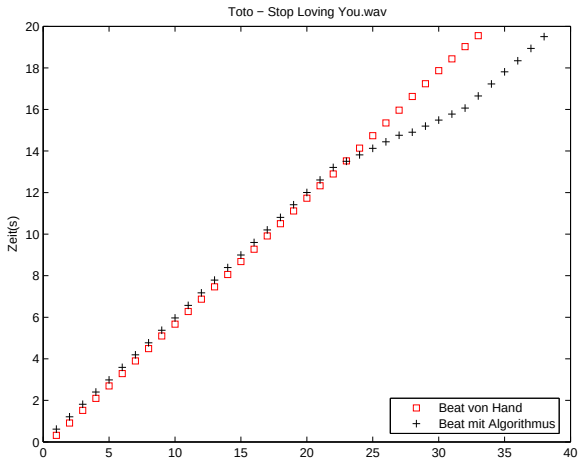

Abbildung 6.2: Vergleich der händisch markierten mit den vom Algorithmus gefundenen Positionen

Obige Abbildung zeigt die Abweichung des detektierten vom wahren Puls vom *on-beat* auf den *off-beat* bei *Stop Loving You* von *Toto*. Dies erklärt sich damit, dass am beginn dieses Stückes die Betonung auf dem *off-beat* liegt. Man kann jedoch nicht von völlig falsch detektierten Positionen sprechen, da trotz der Verschiebung das Tempo stimmt.

Leider konnte das zweite Verfahren zur *beat-detection* bis zum Zeitpunkt der Abgabe nicht mehr ausreichend getestet werden. Tests bei der ISMIR 2007[1] ergaben jedoch, dass beide Verfahren weitgehend gleichwertig sind (Klapuri [14, 15] schnitt etwas besser ab), jedoch war der zeitliche Aufwand für den Algorithmus von Davies [9]um einiges geringer (Ergebnisse aus [18]).

[1] 8th International Conference on Music Information Retrieval, 23. bis 27. September 2007

6.3 Ergebnisse Feature Extraction

Die Auswertung der verschiedenen *feature-extraction*-Methoden gestaltet sich etwas schwieriger, da als Vergleichswert Transkriptionen vorhanden sein müssten, was leider nicht der Fall war. Deshalb können für die einzelnen Ergebnisse nur qualitative Aussagen getroffen werden. Welches der Verfahren für welches Genre bzw. welches Musikstück am besten funktioniert wird sich dann aber in den Ergebnissen des *Thumbailing* zeigen.

Die klarsten und eindeutigeren Ergebnisse zeigen sich für die konstant-Q- Koeffizienten und das *chroma-feature*, etwas undeutlicher sind die Ergebnisse der MFCC's. Nachstehende Abbidlungen zeigen jeweils einen Ausschnitt aus der *feature*-Matrix für Pop2:

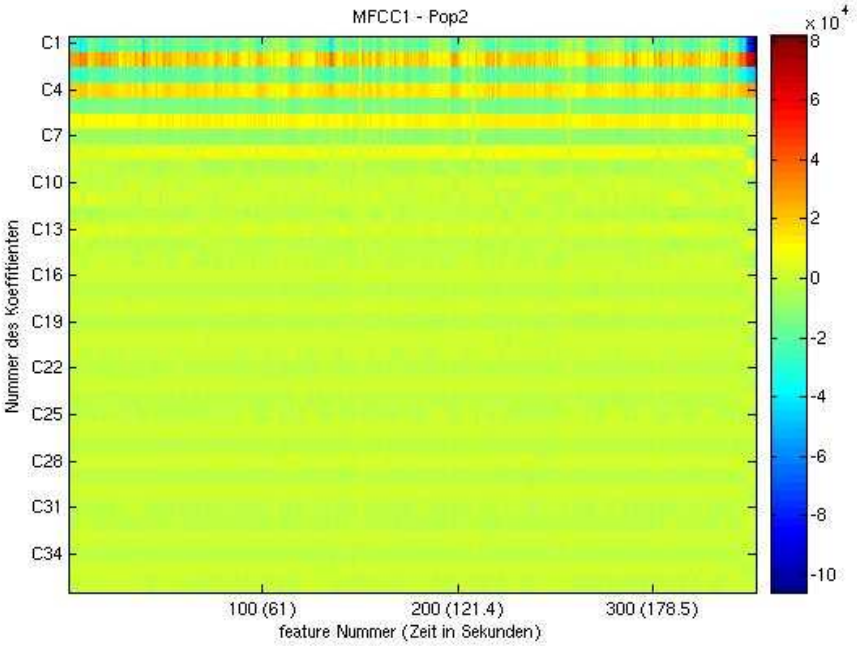

Abbildung 6.3: MFCC's für Pop 2

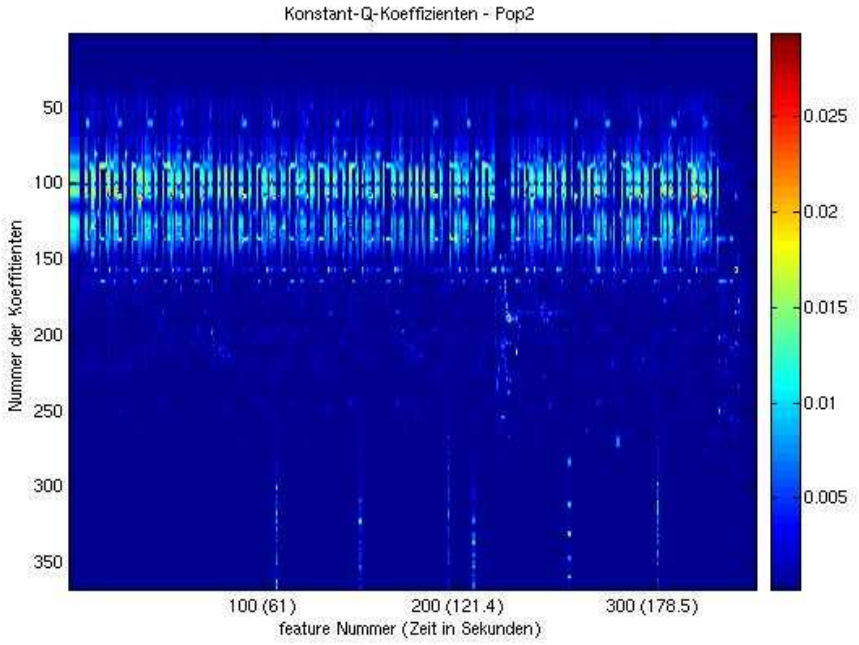

Abbildung 6.5: Konstant-Q-Koeffizienten für Pop 2

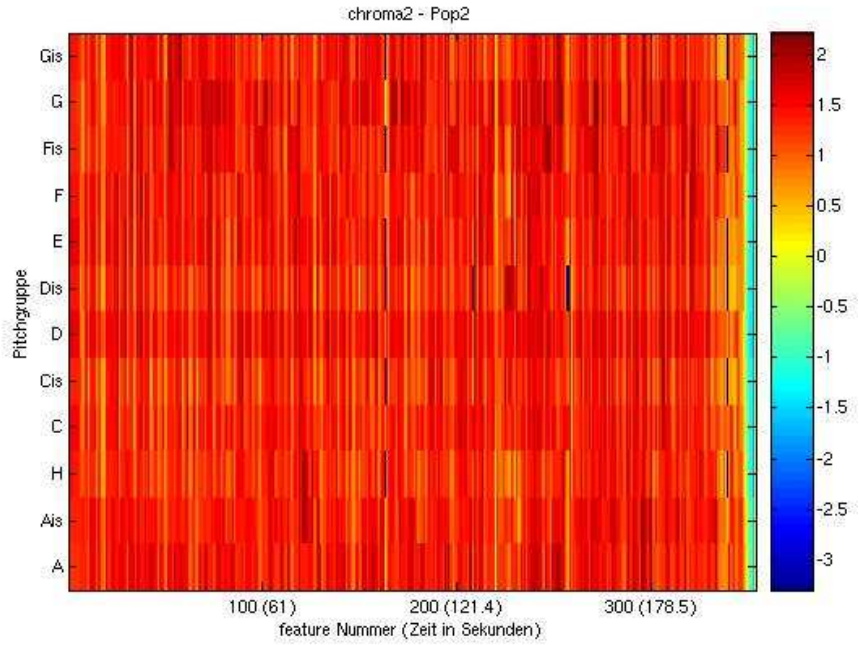

Abbildung 6.4: Chroma für Pop 2

6.4 Ergebnisse Ähnlichkeit

Aus den Ergebnissen der Ähnlichkeitmessung kann bereits eine bessere Aussage über die Funktionalität der *feature-extraction* getroffen werden. Man sieht bereits an der Korrelationsmatrix welche Eigenschaft (*chroma, MFCC, konstant-Q*) sich am besten für das Finden von Wiederholungen eignet. Folgende Tabelle (6.4) zeigt die durch extrahieren der angegebenen *features* gefunden Wiederholungen der jeweiligen Musikstücke, wobei als *frames* die gefundenen *beat*-Punkte verwendet wurden:

Song	***Thumbnail [sek]***	***Wiederholung [sek]***	***Verwendetes feature***
Pop1	*58,7 - 95,6*	*137,5 - 175,6*	*Konst.-Q*
Pop2	*54,2 - 74,0*	*114,5 - 132,9*	*chroma1*
Pop3	*31,8 - 47,0*	*82,8 - 97,5 bzw. 138,9 - 154,3*	*chroma1*
Pop4	*74,6 - 103,0*	*140,5 - 168,2*	*chroma1*
Pop5	*164,9 - 183.9*	*199,8 - 219,4*	*chroma1*
Marsch	*0 - 25,2 bzw. 86,9 - 108,1*	*131,8 - 155,2 bzw. 108,5 -129,5*	*chroma2 bzw. Konst.-Q*
Swing	*72 - 114,5*	*168,2 - 208,6*	*chroma2*
Klassik	*22,5 - 35,0*	*136,0 - 148,3*	*chroma1*

Tabelle 6.4: Gefundene Wiederholungen (*Thumbnails*) in den einzelnen Stücken. Für das Framing wurden *beat*-Positionen verwenden

Die gefundenen Wiederholungen sind dabei durchwegs richtig bzw. sinnvoll für eine Repräsentation des Stückes, also auch als Fingerabdrücke zu verwenden. Da aber bei der Filterung der Matrix nicht über eine konstante Zeit, sonder eine konstante Anzahl von *frames* gefiltert wurde, stimmen in einigen Fällen die Längen von *Thumbnail* und dessen Wiederholung nicht überein. Dies ist darauf zurückzuführen, dass die *beat-detection* nicht immer konsistent für das ganze Stück funktioniert und deshalb *beats* einfügt, welche beim ersten Auftreten des *Fingerprints* nicht als solche detektiert wurden oder umgekehrt

Als *Thumbnails* wurde immer der erste der gefundenen, wiederholten Teile verwendet. Bei den Popsongs war dies immer der Refrain. Bei Pop2 und Pop5 wurden aber auch Mikrostrukturen von jeweils nur ein paar Taktschlägen als wiederholte Teile erkannt. Diese wurden jedoch nicht als Thumbnails verwendet, da sie keine

Song	*Thumbnail [sek]*	*Wiederholung [sek]*	*feature*	*onset-detection*
Pop2	*52,6 - 72,5*	*112,6 - 132,2*	*chroma1*	*SD*
Pop3	*31,0 - 45,4*	*81,5 - 96,5 bzw. 138,5 - 152,1*	*chroma1*	*SD*
Marsch	*0,5 - 25,3*	*133,4 - 157,7*	*chroma1*	*Prob*
Swing	*68,7 - 109,8*	*163,7 - 202,4*	*chroma2*	*HFC*
Klassik	*22,8 - 31,3*	*128,8 - 153,2*	*chroma1*	*LPC*

Tabelle 6.5: Mittels *onset-detection* Gefundene Wiederholungen

richtige Aussage über das Stück brachten, bzw. nicht zum identifizieren des Stückes reichen würden. Solche Mirkostrukturen (jeweils nur einige Sekunden bzw. Taktschläge) könnten jedoch für eine Kodierung verwendet werden, wenn genug sochler (identischen) Teile gefunden werden.

Bei der Verwendung der Onsets für das *framing* war die Ausbeute an *Thumbails* schlechter. Ein Grund dafür ist die meist schlechte Performance der *onset-detection*. Dort, wo *Thumbnails* gefunden wurden, stimmte die zeitliche Position der gefunden Fingerabdrücke in etwa mit jenen aus der *beat*-detection und daher auch mit den wahren Werte überein (siehe nachstehende Tabelle (6.5)).

Für jene Musikstücke, die nicht in Tabelle 6.5 aufgelistetet sind, konnten keine Thumbnails gefunden werden. Dies ist vorallem auf die schlechte Perfomance der *onset-detection* Verfahren zurückzuführen, die die Basis der *feature-extraction* darstellen und großen Einfluss auf deren Performance haben.

Isgesamt kann man zu obigen Ergebnissen sagen, dass diese Methode sich prinzipiell sehr gut eignet, um Ähnlichkeiten in Musikstücken also *Thumbnails* zu finden. Dies zeigt auch ein Vergleich mit der folgenden Tabelle (Tabelle 6.4), in welcher die durch anhören[2] der Musikstücke gefundenen Fingerabdrücke, bzw. deren Positionen im Musikstück aufgelistet sind:

[2]Die eingetragenen Werte sind daher relativ ungenau und dienen nur als Richtwert für die Fingerabdrücke

Song	*Thumbnail [sek]*	*Wiederholung [sek]*
Pop1	*59 - 98*	*138 - 176*
Pop2	*54 - 74*	*114 - 134*
Pop3	*36 - 51*	*87 - 102 bzw. 144 - 160*
Pop4	*75 - 103*	*141 - 169*
Pop5	*165 - 183*	*200 - 219,4*
Marsch	*0 - 25 bzw. 92 - 111*	*133 - 158 bzw. 112-132*
Swing	*72 - 115*	*166 - 208*
Klassik	*22- 34*	*135- 147*

Tabelle 6.6: Tatsächliche Wiederholungen

6.5 Kodierung

Da in allen Fällen zumindest eine Wiederholung gefunden wurde, konnte die angestrebte Kodierung für alle Stücke ausprobiert werden. Da aber nicht alle Wiederholungen sich zu hundert Prozent ähnelten, ergab die Kodierung nur für zwei Stücke ein sinnvolles Ergebnis. Dies waren Pop1, bei dem der zweite Refrain sich nur durch einige kleine Gitarrenverzierungen vom ersten Auftreten des selben unterscheidet. Dadurch ergab sich eine durchaus sinnvolle Kopie des Stückes, nachdem es anhand der Kodierungsvorschrift wieder zusammengefügt wurde. Dazu wurde das Stück in vier Teile unterteilt:

1. Teil 1: 0 - 58,7 Sek (Einleitung & 1. Strophe)
2. Teil 2: 58,7 - 95,6 Sek (Refrain)
3. Teil 3: 95,6 - 137,5 Sek (2. Strophe)
4. Teil 4: 175,6 - 280,8 Sek (Bridge und Schlussteil)

Der zweite Refrain wurde nicht mehr eigens kodiert. Als Dekodierungsvorschrift diente nun nicht mehr die absolute Position des jeweligen Teilstückes, sondern nur mehr die Aneinanderreihung der Teile. Für Pop1 lautet diese *Teil 1, Teil2, Teil3, Teil 2, Teil 4*. In dieser Reihenfolge müssen die Teile aneinandergereiht werden, um wieder das originale Stück zu erhalten (Siehe Abbildung 6.6).

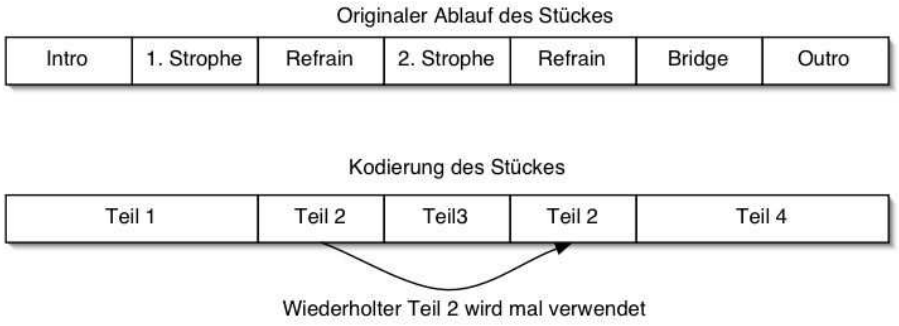

Abbildung 6.6: Kodierung für Pop1

Für das klassische Stück wurde zweimal ein komplett identischer Teil gefunden, welcher dann in der Kodierung nur einmal verwendet werden musste. Dies war der in Tabelle 6.4 aufgezeigte Teil. Bei den restlichen Musikstücken konnte zwar der Sinn des Stückes[3] übermittelt werden, jedoch waren die einzelnen Wiederholungen zu verschieden voneinander, um sie in der Kodierung zu verwenden. Ein Grund dafür war der tatsächliche Unterschied in den einzelnen, wiederholten Teilen (anderer Text, andere Instrumente, usw.), ein weiterer Grund lag darin, dass das timing des *beat*-Positionen nicht immer genau Richitg war, bzw. die schon oben beschriebenen falsch oder nicht detektierten *beats* eine variable Fensterlänge beim Filtern der matrix erfordern würden. Das bedeutet, dass auch wenn die wiederholten Teile gleich lang sind, sie trotzdem eine verschiedene Antzahl von *beats* enthalten können.

[3] Also die Form (AABA, usw.)

Kapitel 7

Zusammenfassung

7.1 Konklusionen

Zusammenfassend kann man sagen, dass eine Kodierung dieser Art durchaus realisiert werden kann, jedoch müssen dabei wirklich zu 100% idente Teile im Musikstück gefunden werden, um das Original wieder herstellen zu können. Zu den einzelnen Schritten, bzw. Teilbereichen ist zu sagen, dass sicherlich die *beat-detection* für das Framing besser geeignet ist, da einfach die Performance besser ist als bei der onset-detection. Zu den extrahierten *features* ist zu sagen, dass die MFCC's sehr schlecht funktionierten und in deren Korrelation fast durchwegs zu große Ähnlichkeit aufwiesen. Sehr gut hingegen funktionierten Chroma und Konstant-Q-Koeffizienten.

Das Finden von Ähnlichkeiten anhand der Filterung der Matrix lieferte ebenso gute wie leicht ablesbare Werte und kann auch in einem Algorithmus durch einfaches Suchen nach lokalen Maxima sehr gut bewerkstelligt werden. Die Hough-Transformation liefert zwar eine anschauliche Darstellung, jedoch ist hier das Auslesen der wiederholten Stellen schwieriger und etwas umständlicher als bei der Filterung der Matrix.

Zum Thumbnailing ist zu sagen, dass aufgrund der konstanten Fensterlänge in der Filterung der Matrix die Thumbails in ihrer Länge und Anzahl der verwendeten Frames genau übereinstimmen müssen, um diese dann in der Kodierung verwenden zu können.

Abschließend bleibt zu sagen, dass mit einigen Verbesserungen dieser Ansatz zur Kodierung sicherlich seine Berechtigung hat.

7.2 Weiterführende Arbeit

Um die Kodierung zu verbessern, muss an erster Stelle die Genauigkeit der *beat-* bzw *onset-detection* erheblich gesteigert werden, da diese das Fundament der *feature-extraction* und somit des ganzen Systems sind. Eine Steigerung der *onset-detecion* könnte eine Verwendung mehrer, verschiedener Ansätze, respektive jener Ansätze, welche die beste Performance brachten, in einem hybriden System bringen. Weiters könnten noch weitere Ansätze getestet werden, die hier leider nichtmehr aufgenommen werden konnten. Vor allm *Zeit-Frequenz Repräsentationen* sowie *Support-Vector-Machines* könnten noch weitere Verbesserungen bringen. Auch das *post-processing* der *onset*-Funktionen könnte noch weiter verbessert werden, vorallen die Auswahl der *peaks* nach der *peak-detection* könnte noch mit einem Wahrscheinlichkeitsmodell verbunden werden, um die Abstände und Positionen besser einzugrenzen. Auch könnte eine Betrachtung verschiedener Toleranzfensterlängen und Untersuchungen bezüglich eines generellen Bias-Offsets (z.B. *onsets* bzw. *beat*-Positionen immer zu früh bzw. zu spät) Aufschluss über Verbesserungsmöglichekeiten geben.

Weiters könnte eine kombinierte *onset-beat-detection* Vorteile bzw. Verbesserungen bringen, in der durch gegenseitiges überwachen der beiden Methoden falsch detektierte Positionen gefunden und ausgemerzt werden könnten. Natürlich sollte auch der Ansatz von Klapuri nochmal gründlich getestet und untersucht werden, der aufgrund einiger Fehler im Matlabcode nicht mehr rechtzeitig fertiggestellt und getestet werden konnte.

Bei der Suche nach Ähnlichkeiten könnte für die Filterung der Korrelationsmatrix eine variable Länge des Filterfensters, welche an die Länge der ersten Wiederholung angepasst ist, verwendet werden. Dazu müsste jedoch eine andere Methode als die Korrelationsmatrix verwendet werden, bzw. aus dieser bereits die Länge des Fingerabruckes berechnet werden können. Weiters könnten die Features kombiniert

werden, auch mit der rhythmischen Struktur, die zwar allein nicht aussagekräftig genug ist, jedoch in Kombination mit den anderen Features sicherlich noch einige Verbesserungen zum Finden von Fingeabdrücken bringen könnte.

Literaturverzeichnis

[1] Samer S. Abdallah and Mark D. Plumbley. *Probability as Metadata:Event Detection in Music Using ICA as a Conditional Density Model.* 4th International Symposium on Independent Component Analysis and Blind Signal Separation (ICA 2003), Nara, Japan, April 2003.

[2] Jean-Julien Aucouturier and Mark Sandler. *Finding Repeating Patterns in Acoustic Musical Signals: Applications for Audio Thumbnailing.* AES 22nd International Conference on Virtual, Synthetic an Entertainment Audio, Espoo, Finland, June 2002.

[3] Mark A. Bartsch and Gregory H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. February 2005. IEEE Transactions on Multimedia Vol. 7, No. 1.

[4] Benjamin Blankertz. The constant q transform.

[5] J.C. Brown. Determination of the meter of musical scores by autokorrelation. *JASA*, 94(4):1953–1957, Oct. 1993.

[6] Judith C. Brown. Calculation of a constant q transform. *JASA*, 89(1):425 – 434, January 1991.

[7] Judith C. Brown. An efficient algorithm for the calculation of a constant q transform. *JASA*, 92(5):2698 – 2701, November 1992.

[8] et. al. Chris Duxbury. A comparison between fixed and multiresolution analysis for onset detection in musical signals. *DAFx*, 2004.

[9] Matthew Davies and Mark Plumbley. Context-dependent beat tracking of musical audio. 15(3), March 2007.

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

[11] Chris Duxbury, Mike Davies, and Mark Sandler. A hybrid arroach to musical onset detection. *DAFx*, September 2002. Chris Duxbury and Mike Davies and Mark Sandler.

[12] Juan Pablo Bello et. al. A tutorial on onset detection in music signals. September 2005.

[13] Udo Zölzer Francois Xavier Nsabimana. Transient encoding of audio signals using dyadic approximations. *DAFx*, September 2007.

[14] Anssi Klapuri. Musical meter estimation and music transcription. 2003.

[15] Anssi Klapuri, Antti Eronen, and Jaakko Astola. Analysis of the meter of acoustic musical signals. 2006.

[16] Pierre Leavau, Laurent Daudet, and Gael Richard. *Methodology and Tools For the Evaluation of Automatic Onset Detection Algorithms in Music.* Paper, 2004.

[17] Sirko Molau. Computing mel-frecuency cepstral coefficients on the power spectrum. Paper.

[18] M. Varewyck. Assessment of state-of-the-art meter analysis systems with an extended meter description model. *ISMIR*, September 2007.

Anhang A

Plots und Tabellen

Tabellen der onset-detection

Jack Johnson - Flake

Methode	TD	TP	TP[%]	FP	FP[%]	FN	FN[%]	TP/(FP+FN)
Prob	70	17	16%	53	76%	88	84%	0,12
LPC	79	55	52%	24	30%	50	48%	0,74
NWPD	100	66	63%	34	34%	39	37%	0,90
WPD	47	29	28%	18	38%	76	72%	0,31
PD	98	66	63%	32	33%	39	37%	0,93
Hyb 3	46	19	18%	27	59%	86	82%	0,17
Hyb 2	21	0	0%	21	100%	105	100%	0,00
Hyb 1	13	1	1%	12	92%	104	99%	0,01
SD	83	59	56%	24	29%	46	44%	0,84
Using Phase	75	47	45%	28	37%	58	55%	0,55
Complex	22	19	18%	3	14%	86	82%	0,21
HFC	47	23	22%	24	51%	82	78%	0,22
Time Domain	42	18	17%	24	57%	87	83%	0,16
# von Hand markierte Onsets:		105						

Tabelle A.1: *Onset-detection* Pop1

Christina Aguilera - Slow Down Baby

Methode	TD	TP	TP[%]	FP	FP[%]	FN	FN[%]	TP/(FP+FN)
Prob	63	13	15%	50	79%	72	85%	0,11
LPC	110	38	45%	72	65%	47	55%	0,32
NWPD	115	38	45%	77	67%	47	55%	0,31
WPD	24	19	22%	5	21%	66	78%	0,27
PD	114	40	47%	74	65%	45	53%	0,34
Hyb 3	54	7	8%	47	87%	78	92%	0,06
Hyb 2	13	0	0%	13	100%	85	100%	0,00
Hyb 1	5	0	0%	5	100%	85	100%	0,00
SD	53	44	52%	9	17%	41	48%	0,88
Using Phase	62	35	41%	27	44%	50	59%	0,45
Complex	48	33	39%	15	31%	52	61%	0,49
HFC	56	15	18%	41	73%	70	82%	0,14
Time Domain	41	7	8%	34	83%	78	92%	0,06
# von Hand markierte Onsets:				85				

Tabelle A.2: *Onset-detection* Pop2

Natalie Imbruglia - Starting Today

Methode	TD	TP	TP[%]	FP	FP[%]	FN	FN[%]	TP/(FP+FN)
Prob	16	6	5%	10	63%	105	95%	0,05
LPC	98	55	50%	43	44%	56	50%	0,56
NWPD	101	63	57%	38	38%	48	43%	0,73
WPD	80	53	48%	27	34%	58	52%	0,62
PD	100	65	59%	35	35%	46	41%	0,80
Hyb 3	23	1	1%	22	96%	110	99%	0,01
Hyb 2	18	0	0%	18	100%	111	100%	0,00
Hyb 1	13	1	1%	12	92%	110	99%	0,01
SD	86	62	56%	24	28%	49	44%	0,85
Using Phase	75	45	41%	30	40%	66	59%	0,47
Complex	69	46	41%	23	33%	65	59%	0,52
HFC	71	13	12%	58	82%	98	88%	0,08
Time Domain	63	13	12%	50	79%	98	88%	0,09
# von Hand markierte Onsets:				111				

Tabelle A.3: *Onset-detection* Pop3

Toto - Stop Loving You

Methode	TD	TP	TP[%]	FP	FP[%]	FN	FN[%]	TP(FP+FN)
Prob	108	24	33%	84	78%	48	67%	0,18
LPC	99	52	72%	47	47%	20	28%	0,78
NWPD	132	47	65%	85	64%	25	35%	0,43
WPD	46	41	57%	5	11%	31	43%	1,14
PD	60	50	69%	10	17%	22	31%	1,56
Hyb 3	169	41	57%	128	76%	31	43%	0,26
Hyb 2	27	5	7%	22	81%	67	93%	0,06
Hyb 1	17	0	0%	17	100%	72	100%	0,00
SD	44	40	56%	4	9%	32	44%	1,11
Using Phase	71	50	69%	21	30%	22	31%	1,16
Complex	44	34	47%	10	23%	38	53%	0,71
HFC	13	3	4%	10	77%	69	96%	0,04
Time Domain	16	7	10%	9	56%	65	90%	0,09
# von Hand markierte Onsets:				72				

Tabelle A.4: *Onset-detection* Pop4

U2 - Wiht or Without You

Methode	TD	TP	TP[%]	FP	FP[%]	FN	FN[%]	TP(FP+FN)
Prob	98	60	43%	38	39%	81	57%	0,50
LPC	77	53	38%	24	31%	88	62%	0,47
NWPD	66	52	37%	14	21%	89	63%	0,50
WPD	91	52	37%	39	43%	89	63%	0,41
PD	63	49	35%	14	22%	92	65%	0,46
Hyb 3	156	71	50%	85	54%	70	50%	0,46
Hyb 2	27	1	1%	26	96%	140	99%	0,01
Hyb 1	41	10	7%	31	76%	131	93%	0,06
SD	72	65	46%	7	10%	76	54%	0,78
Using Phase	133	44	31%	89	67%	97	69%	0,24
Complex	106	39	28%	67	63%	102	72%	0,23
HFC	138	57	40%	81	59%	84	60%	0,35
Time Domain	91	46	33%	45	49%	95	67%	0,33
# von Hand markierte Onsets:				141				

Tabelle A.5: *Onset-detection* Pop5

Frank Sinatra - I've Got You Under My Skin

Methode	TD	TP	TP[%]	FP	FP[%]	FN	FN[%]	TP/(FP+FN)
Prob	127	24	32%	103	81%	51	68%	0,16
LPC	86	23	31%	63	73%	52	69%	0,20
NWPD	113	22	29%	91	81%	53	71%	0,15
WPD	44	9	12%	35	80%	66	88%	0,09
PD	107	17	23%	90	84%	58	77%	0,11
Hyb 3	159	34	45%	125	79%	41	55%	0,20
Hyb 2	8	1	1%	7	88%	74	99%	0,01
Hyb 1	9	2	3%	7	78%	73	97%	0,03
SD	26	15	20%	11	42%	60	80%	0,21
Using Phase	42	9	12%	33	79%	66	88%	0,09
Complex	22	3	4%	19	86%	72	96%	0,03
HFC	39	33	44%	6	15%	42	56%	0,69
Time Domain	35	3	4%	32	91%	72	96%	0,03
# von Hand markierte Onsets:				75				

Tabelle A.6: *Onset-detection* Swing

Johann Strauß - Radetzky Marsch

Methode	TD	TP	TP[%]	FP	FP[%]	FN	FN[%]	TP/(FP+FN)
Prob	136	45	55%	91	67%	37	45%	0,35
LPC	103	10	12%	93	90%	72	88%	0,06
NWPD	105	23	28%	82	78%	59	72%	0,16
WPD	80	17	21%	63	79%	65	79%	0,13
PD	106	27	33%	79	75%	55	67%	0,20
Hyb 3	159	37	45%	122	77%	45	55%	0,22
Hyb 2	14	1	1%	13	93%	81	99%	0,01
Hyb 1	12	1	1%	11	92%	81	99%	0,01
SD	25	9	11%	16	64%	73	89%	0,10
Using Phase	55	9	11%	46	84%	73	89%	0,08
Complex	42	5	6%	37	88%	77	94%	0,04
HFC	57	12	15%	45	79%	70	85%	0,10
Time Domain	55	15	18%	40	73%	67	82%	0,14
# von Hand markierte Onsets:				82				

Tabelle A.7: *Onset-detection* Marsch

Tchaikovsky - Waltz

Methode	TD	TP	TP[%]	FP	FP[%]	FN	FN[%]	TP/(FP+FN)
Prob	73	8	14%	65	89%	48	86%	0,07
LPC	61	11	20%	50	82%	45	80%	0,12
NWPD	107	14	25%	93	87%	42	75%	0,10
WPD	37	0	0%	37	100%	56	100%	0,00
PD	105	14	25%	91	87%	42	75%	0,11
Hyb 3	68	8	14%	60	88%	48	86%	0,07
Hyb 2	1	0	0%	1	100%	56	100%	0,00
Hyb 1	5	0	0%	5	100%	56	100%	0,00
SD	4	2	4%	2	50%	54	96%	0,04
Using Phase	28	4	7%	24	86%	52	93%	0,05
Complex	26	5	9%	21	81%	51	91%	0,07
HFC	102	9	16%	93	91%	47	84%	0,06
Time Domain	55	4	7%	51	93%	52	93%	0,04
# von Hand markierte Onsets:				56				

Tabelle A.8: *Onset-detection* Klassik