
Umsetzbarkeit der Trennung von Percussive und Melodic Frequenzen in einer bereits komprimierten Wav Datei

Projektteil der Belegung einer Wahlspezialisierung
im Studiengang Informatik
an der Fakultät für Informatik und Ingenieurwissenschaften
der Technischen Hochschule Köln

vorgelegt von: Lukas Fey, Nicolas Friedmann
Matrikel-Nr.: 123 456 789, 111 55 463
Adresse: Auf der Platte. 1
51643 Gummersbach
vorname.nachname@smail.th-koeln.de, nicolas-friedmann@gmx.de

eingereicht bei: Prof. Dr. Lutz Köhler
Zweitgutachter*in: Prof. Dr. Vorname Nachname

Ort, TT.MM.JJJJ

Kurzfassung/ *Abstract*

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Inhaltsverzeichnis

Abbildungsverzeichnis	III
1 Einleitung	1
1.1 Relevanz	1
1.2 Vorgehen	2
2 Stand der Wissenschaft	3
3 Hauptteil	4
3.1 Was ist ein Ton?	5
3.2 Formen der Musikdarstellungen	6
3.2.1 Einordnung in den Projektkontext	6
3.3 Aufbau einer Audiodatei/ Music Representation	6
3.3.1 Durchführung der Komprimierung	7
3.3.2 Waveform Audio File Format	8
3.3.3 Andere Formen von Audio-Dateien	8
3.4 Trennung einer Tonspur in verschiedene Instrumente	8
3.4.1 Fourier Transform	9
3.4.2 Wavelet Transform	11
3.4.3 Wahl der Fourier Transform	12
3.4.4 Short-Time Fourier Transform	12
4 Code	13
4.1 Beschreibung des Codes	13
4.2 Hauptfunktion: <code>harmonic_extraction</code>	13
4.3 Speichern der Ergebnisse	14
4.4 Fourier-Transformation und <code>librosa</code>	14
4.5 Hintergrund: Fourier-Transformation und HPSS in <code>librosa</code>	15
4.5.1 Short-Time Fourier Transform (STFT)	15
4.5.2 Harmonic-Percussive Source Separation (HPSS)	16
4.5.3 Inverse STFT (iSTFT)	16
Literatur	18
Anhang	20

Abbildungsverzeichnis

3.1	Wellenform eines Tons	5
3.2	Unterteilung in Samplingrate (oben) und Quantisierungsschrittweite (unten)	7
3.3	Note C4 in unterschiedlichen Darstellungen (Müller 2021, S. 41) . . .	10

1 Einleitung

Musik ist zu einem Teil des täglichen Lebens vieler Menschen geworden. Durch den Konsum und den wirtschaftlichen Ertrag wird an der Produktion und Analyse von Musik geforscht (Quelle?).

Ein Teil der Forschung bezieht sich auf die Trennung einer Wellenform in die unterschiedlichen Funktionen der Frequenzen. Zuvor beinhaltet die Wellenform eine oder mehrere Sinus- und Kosinusfunktionen, die eine neue Funktion bilden. Dies wird sowohl auf audiovisuelle als auch auf visuelle Funktionen angewendet. Beispielsweise kann die Funktion mehrere Instrumente beinhalten, die anhand der Funktion nicht identifizierbar sind. Einer der Algorithmen zur Trennung von Signalen heißt Fourier Transform und wird in diesem Projekt behandelt.

Unter anderem werden Musikinstrumente in Liedern getrennt und einzeln angehört oder wiederverwendet. In diesem Projekt werden zum Einstieg lediglich perkussive aus akkustischen Instrumenten getrennt. Zwischen den jeweiligen Instrumentengruppen wird in Abschnitt 3.1 unterschieden.

1.1 Relevanz

Es gibt unterschiedliche Kontexte, in denen die Trennung von Audiosignalen zum Einsatz kommt. Häufig ist eine Eingabefunktion als Summe aller Signale schwierig zu analysieren oder weiterzuverarbeiten. Beispielsweise, wenn einzelne Instrumente im Nachhinein bearbeitet werden sollen.

In Zeiten von zunehmend digital produzierter Musik und Verbreitung von Informatik stellt sich die Frage, wie Musiksignale digital aufgebaut sind und wie man mit ihnen arbeiten kann. Im Kontext dieser Arbeit wird die Fourier Transform verwendet, um eine Tondatei in Gruppen von harmonischen und perkussiven Musikinstrumenten zu zerlegen. Dies kann mit weiterer Modifikation verwendet werden, um bestimmte Instrumente zum üben zu trennen oder Hintergrundgeräusche auszublenden.

(Etwas zu kurz?)

1.2 Vorgehen

Ohne jegliches Vorwissen, wie die Signale einer Musikdatei gespeichert und von einem Computer interpretiert oder getrennt werden, wird das Projekt durchgeführt. Die Erkenntnisse und notwendiges Hintergrundwissen werden dokumentiert und erklären die Logik und Zusammenhänge der Fourier Transform. Durch die Dokumentation wird anderen Leser:innen ein erstes Verständnis von WAV-Dateien vermittelt auf dem im Anschluss aufgebaut werden kann.

(WAV-Dateien sehr spezifisch: Den Leser:innen wird auch die Funktionalität der FFT und Code etc. vermittelt?)

Das Projekt dokumentiert anschließend die Implementierung eines Algorithmus zur Trennung der harmonischen und perkussiven Tongruppen einer WAV-Datei mittels Fourier Transform, sowie das schreiben auf zwei getrennte WAV-Dateien mittels Inverse Fourier Transform (siehe: Kapitel 4). Dabei wird zuvor behandelte Logik und Hintergrundwissen referenziert und in der Praxis erklärt.

2 Stand der Wissenschaft

Die Trennung von Musikinstrumenten ist ein praktischer Anwendungsfall zur Separation von Signalen. Dabei liegen in einer Aufnahme mehrere Signale in Form von einer Funktion vor. In dieser Funktion sind die einzelnen Signale schwierig zu identifizieren.

Bei Vorgehen zur Trennung von Signalen wird die Funktion in eine Abhängigkeit zur Frequenz umgeformt. Anhand der Frequenz können unterschiedliche Signale identifiziert werden. Beispielsweise werden die Frequenzen unterschiedlichen Instrumenten zugeordnet.

Inzwischen wurden mehrere Möglichkeiten zur Transformation einer Funktion in Abhängigkeit zur Zeit in die Abhängigkeit zur Frequenz entdeckt. Diese unterscheiden sich in der Darstellung der Funktion und der Genauigkeit des Ergebnisses.

In diesem Projekt wird die Fourier Transform angewendet, die verbreitet und sehr effektiv ist (Quelle?). Es wird überprüft, ob die Implementierung und Anwendung ohne Vorkenntnisse und anspruchsvolle Hardware umsetzbar ist.

Eine weitere Methode ist die Wavelet-Transformation. Diese wird in Abschnitt 3.4.2 erläutert, um die Entscheidung für die Fourier Transform zu begründen.

3 Hauptteil

Um ein grundlegendes Verständnis für die Analyse von Dateiformaten zu entwickeln wird einem klaren Schema gefolgt. Mit diesem Schema wird sich an der chronologischen Reihenfolge von der Entstehung eines Tons bis zur Verarbeitung im Code orientiert.

1. Was ist ein Ton
2. Formen der Musikdarstellung
3. Vorstellung der Fourier Transform
4. Alternative Methode zur Fourier Transform
5. Trennung von Instrumenten im Code
6. Vorgehen im Projekt (evtl früher)
7. Wie kann man auf dem Stand aufbauen? (/Wie könnte es weitergehen?)
8. Fazit
 - a) Aufwand
 - b) Ertrag (z.B. erworbenes Wissen)

Um einen Algorithmus zu implementieren, ist es hilfreich die unterschiedlichen Musikdarstellungen kennenzulernen. Für eine effektive Fourier Transform gibt es Voraussetzungen, die nur von wenigen Musikdarstellungen erfüllt werden.

Außerdem ist es relevant ein Verständnis für die Fourier-Transformation und andere Methoden zu entwickeln, um nachzuvollziehen warum die Fourier Transform geeignet ist. Zudem hilft ein Verständnis bei der Implementierung des Codes.

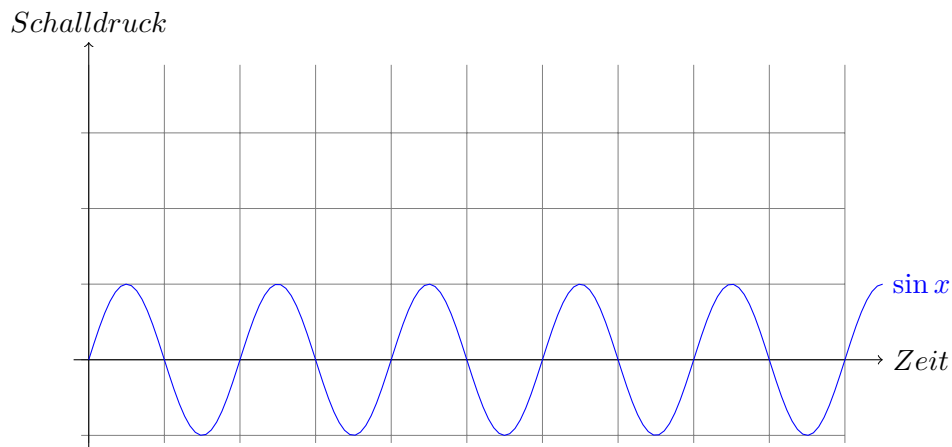


Abbildung 3.1: Wellenform eines Tons

3.1 Was ist ein Ton?

Töne entstehen durch die Vibration eines Gegenstandes. Diese erzeugen Schallwellen. Das Gehör kann diese Schwingungen der Luft wahrnehmen (Auersignal o. D.). Der Luftdruck einer Schallwelle wird graphisch als eine Sinus- oder Cosinusfunktion dargestellt.

Anhand der Wellenform kann die Frequenz eines Tons abgeleitet werden. Diese wird in Hertz (kurz: Hz) angegeben. Die Frequenz gibt die Anzahl an Zyklen des Tons pro Sekunde an. Wenn beispielsweise Abbildung 3.1 eine Sekunde darstellt, entspricht die Frequenz 53Hz .

Diese Vibration kann durch unterschiedliche Gegenstände erzeugt werden. Dazu zählen Becken eines Schlagzeugs, Saiten eines Kontrabasses oder die Stimmbänder. In diesem Projekt werden die Perkussionsinstrumente von den restlichen Instrumenten einer Wav-Datei getrennt.

Akkustische Instrumente erzeugen einen Ton indem Menschen Kraft auf sie ausüben. Sie benötigen keinen Strom, keine Verstärkung und keinen Computerprozessor, um einen Ton zu erzeugen und sind die ältesten Instrumente. Sie können in perkussive und harmonische Instrumente unterteilt werden. Perkussive Instrumente werden geschlagen, geschüttelt oder geschabt und sind eine spezielle Form der akkustischen Instrumente. Die übrigen akkustischen Instrumente sind harmonisch (Dr. Victor Ezquerro 2020).

Neben akkustischen existieren noch elektrische und digitale Instrumente. Sie benötigen entweder Elektrizität oder einen Computerprozessor, um wie vorgesehen zu funktionieren. Allerdings werden sie in diesem Projekt nicht behandelt.

3.2 Formen der Musikdarstellungen

Musik kann unterschiedlich dargestellt werden. Je nach Bedarf, wird eine andere Form der Musikdarstellung benötigt. Es werden Musiknoten, symbolische Darstellungen und Audiodarstellungen verwendet. Es ist eine formale Sprache, die vorgibt wie ein Musikstück gespielt wird (Meinard Müller and Vlori Arifi-Müller o. D.). Bei symbolischen Darstellungen werden eindeutige Entitäten definiert, die von einem Computer übersetzt werden. Beispielsweise wird der Musical Instrument Digital Interface (kurz: MIDI) Standard verwendet, um Informationen eines gespielten Tons möglichst detailliert zu speichern und abzurufen.

Eine weitere Form der Darstellung, ist die Audiodarstellung. In diesen Darstellungen werden alle Informationen der Töne als Audiosignale digital gespeichert und geteilt. Es werden die Schallwellen eines Tons (siehe Abschnitt 3.1) aufgenommen und digital als eine Wellenform gespeichert, die an der Schallwelle orientiert ist. Dazu gehören auch das Timing, die Intensität, die Lautstärke, die Länge des Tons und vieles mehr. Es werden nicht die einzelnen Töne und Noten gespeichert, sondern die Frequenzen während der Aufnahme in Abhängigkeit zur Zeit. Die Audiodatei kann auch Nebengeräusche oder weitere Instrumente beibehalten (Müller 2021, S. 1ff).

3.2.1 Einordnung in den Projektkontext

Die digitale Darstellung macht es schwieriger unterschiedliche Audiosignale zu trennen und die ursprünglichen Töne wieder herzustellen. Die verbreitetste Form der Audiodarstellung ist MP3 (NRVision o. D.). In diesem Projekt werden jedoch Wav(einheitlich?)-Dateien behandelt. Die Unterschiede, sowie Vor- und Nachteile der Audiodarstellungen werden in Abschnitt 3.3.3 behandelt (Müller 2021).

3.3 Aufbau einer Audiodatei/ Music Representation

Für die Aufnahme von Audiosignalen werden analoge Signale in eine digitale Form von Schall umgewandelt. Die analoge Signaldarstellung basiert auf kontinuierliche und konstante Spannungsschwankungen, die den verursachten Luftdruckschwankungen entsprechen (Electronic Music Interactive o. D.). In der digitalen Form werden

die Spannungsschwankungen als Bitstreams gespeichert und je nach Audio-Format komprimiert.

3.3.1 Durchführung der Komprimierung

Die Komprimierung von Audiodateien wird meistens verwendet, um mehr Musik auf einem Datenträger speichern zu können (Audioengine o. D.). Um eine Aufnahme zu komprimieren wird diese in eine Samplingrate (auch: Abtastrate) und eine Quantisierungsschrittweite reduziert.

Samplingrate

Die Samplingrate definiert die Anzahl von gespeicherten Signalen pro Sekunde. Dies reduziert die benötigten gespeicherten Daten von einer durchgängigen Aufnahme auf mehrere Blöcke. Beispielsweise hat eine CD Aufnahme eine Samplingrate von 44.100 Samples pro Sekunde (kurz: 44.1 kHz), das heißt es werden 44.100 Soundsignale pro Sekunde gespeichert, dessen Übergänge kaum wahrnehmbar sind, jedoch bereits zu einem deutlichen Reduzierung des Speicherbedarfs führen.

Quantisierungsschrittweite

Die Quantisierungsschrittweite beschreibt die Auflösung, mit der jedes Sample komprimiert wird. Die möglichen Werte eines Samples sind kontinuierlich und könnten theoretisch unendlich viele Dezimalstellen haben. Durch die Quantisierungsschrittweite werden diese Werte jedoch auf eine festgelegte Anzahl möglicher diskreter Werte reduziert, wodurch der Unterschied zwischen den Werten bestimmt wird. Bei CDs wird beispielsweise eine 16-Bit-Codierung gewählt, die 65.536 mögliche Werte definiert.

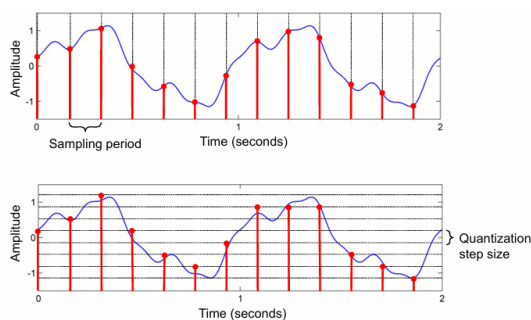


Abbildung 3.2: Unterteilung in Samplingrate (oben) und Quantisierungsschrittweite (unten)

3.3.2 Waveform Audio File Format

Das Waveform Audio File Format (kurz: WAV) speichert Audioaufnahmen unkomprimiert (Roxio o. D.). Der Begriff leitet sich „vom englischen Wort ‚wave‘“ für Schallwelle ab (e-teaching.org o. D.). Für die in diesem Projekt behandelte Fourier Transform sind insbesondere die „Abtastrate des Messsystems“ und die Quantisierungsschrittweite (siehe Abschnitt 3.3.1) entscheidend (NTi Audio o. D.). Aufgrund der unkomprimierten Speicherung (und der damit verbundenen hohen Abtastrate und Quantisierungsschrittweite) lässt sich eine klare Trennung von Tonspuren ermöglichen (Referenz?).

3.3.3 Andere Formen von Audio-Dateien

- MP3
- WMA
- AAC
- OGG
- FLAC
- RM

(Lehrerinnenfortbildung Baden-Württemberg o. D.)

3.4 Trennung einer Tonspur in verschiedene Instrumente

Eine Aufnahme speichert das aufgenommene Signal in Abhängigkeit zur Zeit. Jedes eingehende Signal verfügt über Schwingungen in Form von einer Wellenform. Bei einer Aufnahme vermischen sich die verschiedenen Signale zu einer gemeinsamen Wellenform und sind daher schwierig voneinander zu unterscheiden, beispielsweise bei Hintergrundgeräuschen oder der gleichzeitigen Aufnahme mehrerer Musikinstrumente.

Die Trennung von Instrumenten innerhalb einer Tonspur ist ein Thema, das in Forschung und Bildung intensiv behandelt wird (Quelle?). Die am häufigsten verwendete Methode ist die Fourier Transform (Quelle?), die es ermöglicht, ein Signal in seine Frequenzkomponenten zu zerlegen.

Neben der Fourier Transform werden auch andere Ansätze wie die WaveletbTransform, die Short-Time Fourier Transform (kurz: STFT), sowie statistische Verfahren wie

die Blind Source Separation (BSS) und die Independent Component Analysis (ICA) eingesetzt. Darüber hinaus finden moderne Verfahren des maschinellen Lernens, insbesondere neuronale Netzwerke, zunehmend Anwendung in der Trennung von Audiosignalen.

3.4.1 Fourier Transform

Die Fourier Transform ist ein Algorithmus der die Darstellung einer Tondatei verändert. Ursprünglich liegt die Audiospur mit einer Kombination aus unterschiedlichen Frequenzen in Abhängigkeit zur Zeit vor. In dieser Darstellung sind die unterschiedlichen Signale schwierig zu trennen und werden von der Fourier Transform transformiert.

Entwicklung der Fourier Transform

Die Fourier Transform ist eine Verallgemeinerung der Fourierreihen. Diese Reihen können stetige oder stückweise stetige Funktionen in eine Summe von Sinus- und Kosinusfunktionen zerlegen. Bereits im 18. Jahrhundert wurden Fourierreihen für spezifische Funktionen entdeckt. 1822 stellte Joseph Fourier die Hypothese auf, dass sich jede Funktion als Summe solcher Reihen darstellen lässt. Erst im 20. Jahrhundert wurden Fourierreihen auch für andere stetige oder stückweise stetige Funktionen formal bewiesen. Dank der Vollständigkeit der Funktionenreihe lässt sich die Fourier Transform auf eine Vielzahl von Funktionen anwenden, einschließlich periodischer und nicht-periodischer Funktionen, und erhielt ihren Namen zu Ehren von Fourier.

Durchführung der Transformation

Die Fourier Transform ist ein mathematisches Verfahren, bei dem ein Signal aus dem Zeitbereich in den Frequenzbereich transformiert wird. Die Transformation ermöglicht es, beliebige periodische und stückweise stetige Funktionen als Summe von Sinus- und Kosinuswellen unterschiedlicher Frequenzen darzustellen.

In der neuen Darstellung werden die Frequenzen der Funktion unabhängig von der zeitlichen Komponente wiedergegeben. Unterschiedliche Frequenzen können unterschiedlichen Signalen zugeordnet werden. Die frequentielle Darstellung gibt an welche Signale in welchen Frequenzen Teil der Funktion sind, allerdings nicht wann. Daher wird die Darstellung wieder umgeformt in die zeitliche Abhängigkeit.

Das Signal oder die Signale, die von der Tonspur getrennt werden, können in der frequentiellen Darstellung identifiziert werden. Anschließend werden diese von den übrigen Signalen getrennt und zurück in die ursprüngliche Darstellung transformiert

(Inverse Fourier Transform). Damit erhält man eine neue Tondatei, die ausschließlich aus den benötigten Signalen besteht.

Durchführung am Beispiel einer Musiknote

In diesem Beispiel (aus Müller 2021, S.40f) wird eine Note auf einem Piano gespielt und durch die Transformation in eine frequentielle Darstellung der gespielte Ton erkannt.

Die Aufnahme des Tons ist in Abbildung 3.3(a) zu erkennen. Für die Transformation wird ein Ausschnitt von 10ms verwendet, um den Rechenaufwand zu reduzieren.

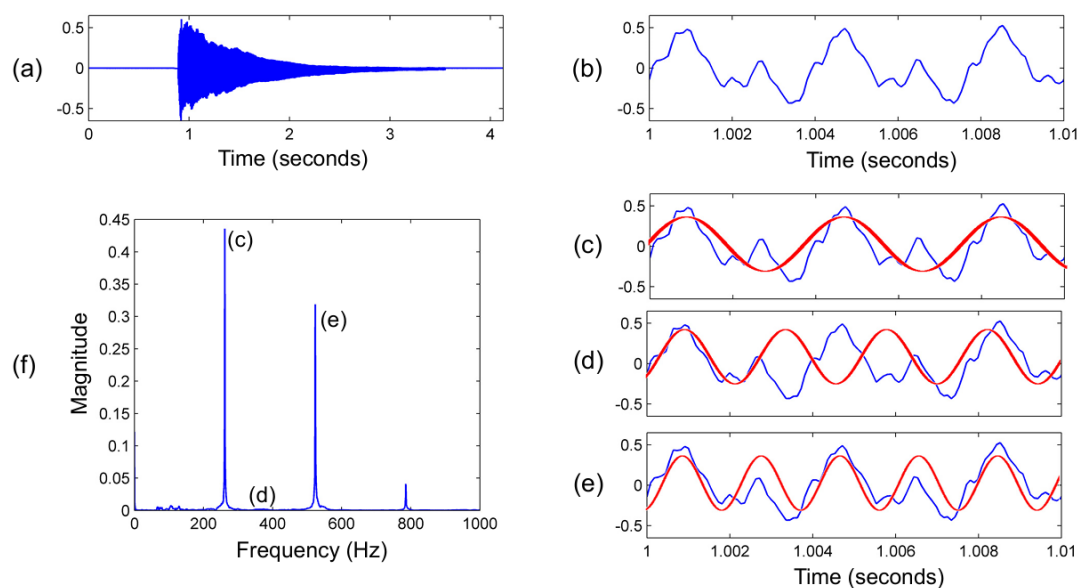


Abbildung 3.3: Note C4 in unterschiedlichen Darstellungen (Müller 2021, S. 41)

Anschließend werden unterschiedliche Vergleichsfunktion für die jeweiligen Frequenzen mit dem Ausschnitt der Tonspur verglichen. Die Ähnlichkeiten der jeweiligen Frequenzen werden in (f) wiedergegeben.

In Abbildung 3.3(c) ist die Übereinstimmung für die Frequenz $w = 262$ Hz besonders hoch. Daraus folgt in (f) bei ungefähr 262 der höchste Wert. Die Höhe des Wertes wird in der Variable dw angegeben.

Die Frequenz 262 entspricht der Note C4. Darüberhinaus wird bei einer Frequenz von 523 (siehe Abbildung 3.3(e)) eine hohe Übereinstimmung erkannt. Dies entspricht ungefähr der Frequenz des zweiten Teils der Note C4 (zweiter Teil?).

Nachteile

Die Fourier Transform ermöglicht es je nach Bedarf zwischen der zeitlichen oder der sequentiellen Darstellung zu wechseln. Allerdings ist bei der Fourier Transform der Wechsel zwischen den Darstellung notwendig und es wird entweder die zeitliche oder die frequentielle Komponente ignoriert. Bei der Anwendung der Fourier Transform gibt es keine Darstellung die beide Komponenten kombiniert.

- Darstellung (nie Zeit und Frequenz)

(Noch unklar wie unterschiedliche Instrumente vorliegen, aber analoge Instrumente sollten in Sinusfunktion vorliegen und erkennbar sein) (Lukas: Alle Instrumente bzw Töne der Instrumente liegen als Sinuswellen vor)

3.4.2 Wavelet Transform

Die Wavelet Transform ist ein Verfahren, das eine zeitliche Darstellung einer Funktion in eine dreidimensionale Darstellung in Abhängigkeit von Zeit und Frequenz überführt. Dabei werden sogenannte Wavelets - spezielle Wellenfunktionen - mit der ursprünglichen Funktion verglichen, um Übereinstimmungen zu finden. Der Begriff „Wavelet“ stammt aus dem Französischen und bedeutet „kleine Welle“ oder „Wellchen“.

Im Gegensatz zur ursprünglichen Funktion haben die Wavelets eine endliche Fläche (auch: finite energy), was sie begrenzt und lokalisiert. Eine weitere Bedingung für Wavelets ist, dass ihr Integral gleich null ergibt, d.h., dass die Fläche über und unter der X-Achse gleich groß ist (Admissibility condition).

Jedes Wavelet wird durch die Parameter m und b ergänzt:

m : Bestimmt die Frequenz des Wavelets

b : Bestimmt den Zeitpunkt des Wavelets

Zudem besitzt das Wavelet einen realen und einen imaginären Teil. Durch die Berücksichtigung des imaginären Teils entsteht eine dreidimensionale Darstellung des Signals. Bei der Wavelet Transform wird sowohl der reale als auch der imaginäre Teil des Wavelets mit der Funktion korreliert (Mathematik: Korrelation), um die Ähnlichkeit der Funktion und des Wavelets zu berechnen. Diese Ähnlichkeit wird für jedes m und b ermittelt und in einem dreidimensionalen Ausgabe-Graphen in Abhängigkeit von der Zeit dargestellt. Dadurch ergibt sich eine Darstellung des Signals in Bezug auf Zeit und Frequenz.

Ein Anwendungsfall ist die Überprüfung von Ampelleuchten. Während die Fourier-Transformation die verschiedenen Frequenzen der Farben Grün, Gelb und Rot erkennen kann, um festzustellen, ob die Lampen leuchten, erlaubt die Wavelet Transform zusätzlich die Angabe, ob die Lichter zu den richtigen Zeitpunkten aufleuchten (Artem Kirsanov 2022).

- verwendet spezialisierte Funktionen: Wavelets - aus französisch Wellchen/ kleine Welle - Familie an Funktionen - jede für spezielle Anwendung - Integral unter und über X-Achse gleich groß - Admissibility condition - endliche Fläche - finite energy - lokalisiert in Zeit

3.4.3 Wahl der Fourier Transform

Für die Trennung der Instrumente einer Wav-Datei wurde die Fourier Transform gewählt, trotz einiger Alternativen. Die Fourier Transform ist eins der meistverwendeten Werkzeuge der Signalverarbeitung (Müller Kap. 2). Der größte Nachteil der Fourier Transform ist, dass nicht gleichzeitig die Zeit und die Frequenz Domäne dargestellt werden können. Allerdings reicht dies bei der Trennung von Musikinstrumenten, da die Darstellung zum Schluss wieder in die zeitliche Domäne umgeformt werden soll, um das Ergebnis in eine Wav-Datei zu überführen. Außerdem verliert die Fourier Transform wenig Informationen durch die klare Trennung von zeitlicher und frequentieller Darstellung (<https://academic.oup.com/jmammal/article-abstract/81/4/927/2372896>). Allerdings ist die Fourier Transform bei größeren Dateien aufwändiger und fehleranfälliger, falls die ganze Datei transformiert wird (auch zu Nachteile?). Jedoch wird diese Limitierung durch die Verwendung der Short-Time Fourier Transform reduziert (<https://www.sciencedirect.com/science/article/abs/pii/S1746809416301859>: 6.1.3.1).

3.4.4 Short-Time Fourier Transform

Die Short-Time Fourier Transform (auch: STFT) basiert auf der Fourier Transform, dessen Nachteile zunehmender Aufwand und fehleranfälligkeit bei der Transformation größerer Dateien beinhaltet (siehe: Nachteile oder Wahl?).

Stattdessen teilt die Short Time Fourier Transform eine Datei in mehrere kleine Pakete, deren Transformation effizienter und effektiver durchgeführt werden können. Anschließend werden die Übergänge der Pakete geglättet (Wortwahl?), um unregelmäßigkeiten (groß?) zu vermeiden. In Abschnitt 4.5.1 wird die praktische Umsetzung mittels Programmcode dargestellt und die Funktionalität erläutert.

4 Code

In diesem Abschnitt wird der Python-Code zur Extraktion von harmonischen und perkussiven Komponenten aus Audiodateien unter Verwendung der Bibliothek `librosa` detailliert erklärt.

4.1 Beschreibung des Codes

Der Code beginnt mit dem Importieren der benötigten Bibliotheken, um Audio zu laden, zu verarbeiten und in neue Dateien zu schreiben:

Listing 4.1: Bibliotheken importieren

```
import os
import librosa
import soundfile as sf
import numpy as np
```

Hierbei ist `librosa` die zentrale Bibliothek zur Verarbeitung von Audiodaten, während `soundfile` für das Schreiben der resultierenden Audiodateien verwendet wird.

4.2 Hauptfunktion: `harmonic_extraction`

Die Hauptfunktion des Codes ist `harmonic_extraction`, die einen Dateipfad für die Audiodatei und einen Namen für die Ausgabedatei als Parameter erhält.

Listing 4.2: Die Funktion `harmonic_extraction`

```
def harmonic_extraction(audio_path, output_filename):
    y, sr = librosa.load(audio_path)

    y_harmonic, y_percussive = librosa.effects.hpss(y)
```

Die Funktion beginnt mit dem Laden der Audiodatei. Die Methode `librosa.load` lädt die Datei und gibt das Audiosignal `y` und die Sampling-Rate `sr` zurück. Der nächste

Schritt ist die Verwendung der Harmonic-Percussive Source Separation (HPSS) Funktion `librosa.effects.hpss(y)`, die das Audiosignal in harmonische und perkussive Komponenten zerlegt. Dies geschieht mithilfe der Fourier-Transformation.

4.3 Speichern der Ergebnisse

Nach der Zerlegung speichert der Code die beiden Komponenten in separaten Dateien:

Listing 4.3: Speichern der Komponenten

```
current_dir = os.getcwd()

input_audio_path = os.path.join(current_dir, 'audios', audio_path)
output_directory = os.path.join(current_dir, 'extractedfiles')
output_path_harmonic = os.path.join(output_directory, output_filename)
output_path_percussive = os.path.join(output_directory, 'extracted_percussive.wav')

os.makedirs(output_directory, exist_ok=True)

sf.write(output_path_harmonic, y_harmonic, sr)
print("Harmonic_component_saved_to:", output_path_harmonic)

y_percussive_only = y - y_harmonic

sf.write(output_path_percussive, y_percussive_only, sr)
print("Percussive_component_saved_to:", output_path_percussive)
```

Das obige Codefragment erstellt das Verzeichnis **extractedfiles** und speichert darin die harmonischen und perkussiven Komponenten als separate Dateien. Die Methode `sf.write` schreibt das Audiosignal in eine Datei, die anschließend abgespielt oder analysiert werden kann.

4.4 Fourier-Transformation und librosa

Im Hintergrund verwendet **librosa** die Fourier-Transformation zur Analyse und Trennung der Frequenzkomponenten. Die Fourier-Transformation wandelt ein Zeitsignal in seine Frequenzkomponenten um, was es **librosa** ermöglicht, harmonische und perkussive Elemente zu isolieren.

Die Harmonic-Percussive Source Separation (HPSS) Methode analysiert das Spektrum des Audiosignals und trennt es basierend auf der Stabilität der Frequenzkomponenten

über die Zeit: Harmonische Komponenten bleiben relativ konstant, während perkussive Komponenten abrupte Änderungen im Spektrum aufweisen.

4.5 Hintergrund: Fourier-Transformation und HPSS in librosa

Um die Funktionsweise von `librosa` bei der Verarbeitung und Trennung von Audiosignalen nachzuvollziehen, betrachten wir die verwendeten Algorithmen, insbesondere die Short-Time Fourier Transform (STFT) und die Harmonic-Percussive Source Separation (HPSS). Diese Verfahren lassen sich mithilfe von grundlegender Signalverarbeitung in Python umsetzen.

4.5.1 Short-Time Fourier Transform (STFT)

Die STFT teilt das Audiosignal in kurze, sich überlappende Abschnitte, um das zeitliche Verhalten der Frequenzanteile zu erfassen. Dies ergibt ein Spektrogramm, das Frequenzänderungen im Zeitverlauf darstellt. Der Code für die STFT ist wie folgt:

Listing 4.4: STFT-Implementierung

```
import numpy as np

def stft(y, n_fft=2048, hop_length=512, window="hann"):
    if window == "hann":
        win = np.hanning(n_fft)
        num_frames = 1 + (len(y) - n_fft) // hop_length
        stft_matrix = np.empty((n_fft // 2 + 1, num_frames), dtype=np.complex64)

        for i in range(num_frames):
            start = i * hop_length
            frame = y[start : start + n_fft] * win
            stft_matrix[:, i] = np.fft.rfft(frame)

    return stft_matrix
```

Dieser Code berechnet das Spektrogramm des Audiosignals, indem für jedes Segment die Fourier-Transformation mit `np.fft.rfft` durchgeführt wird. Das Hann-Fenster glättet die Segmente und reduziert abrupte Übergänge zwischen den Abschnitten.

4.5.2 Harmonic-Percussive Source Separation (HPSS)

Die HPSS-Technik trennt das Spektrum des Audiosignals in harmonische und perkussive Komponenten, basierend auf der Annahme, dass harmonische Frequenzen über die Zeit stabil bleiben, während perkussive Frequenzen abrupte Änderungen aufweisen. Hierfür werden Medianfilter verwendet:

Listing 4.5: HPSS-Implementierung

```
from scipy.ndimage import median_filter

def harmonic_percussive_separation(stft_matrix, harmonic_filter_size=31, percussive_filter_size=31):
    harmonic_component = median_filter(np.abs(stft_matrix), size=(1, harmonic_filter_size))
    percussive_component = median_filter(np.abs(stft_matrix), size=(percussive_filter_size, 1))

    harmonic_mask = harmonic_component > percussive_component
    percussive_mask = percussive_component >= harmonic_component

    harmonic_stft = stft_matrix * harmonic_mask
    percussive_stft = stft_matrix * percussive_mask

    return harmonic_stft, percussive_stft
```

Dieser Code verwendet `median_filter`, um das Spektrum über die Zeitachse zu glätten und so harmonische und perkussive Komponenten zu trennen. Die beiden Masken trennen das Spektrum in harmonische und perkussive Anteile und erlauben eine gezielte Extraktion der einzelnen Bestandteile.

4.5.3 Inverse STFT (iSTFT)

Um das transformierte Spektrum wieder in ein Zeitsignal zu konvertieren, verwenden wir die inverse STFT:

Listing 4.6: iSTFT-Implementierung

```
def istft(stft_matrix, hop_length=512, n_fft=2048, window="hann"):
    if window == "hann":
        win = np.hanning(n_fft)
    y = np.zeros(hop_length * (stft_matrix.shape[1] - 1) + n_fft)

    for i in range(stft_matrix.shape[1]):
        frame = np.fft.irfft(stft_matrix[:, i]) * win
        start = i * hop_length
        y[start : start + n_fft] += frame

    return y
```

Die Funktion `istft` führt das Frequenzspektrum zurück in die Zeitdomäne und rekonstruiert das Audiosignal durch eine inverse Fourier-Transformation mit `np.fft.irfft`. Die Überlappungsaddierung gewährleistet eine nahtlose Rücktransformation in die Zeitdomäne.

Zusammenfassend lässt sich sagen, dass `librosa` mithilfe der STFT und HPSS das Audiosignal in harmonische und perkussive Anteile zerlegt und so die Frequenzkomponenten eines Signals analysieren und trennen kann.

Literatur

- Artem Kirsanov (Aug. 2022). *Wavelets: a mathematical microscope*. Abgerufen am 19.08.2024. URL: <https://youtu.be/jnxqHcObNK4?si=1VZPZquUYXSuI4vN>.
- Ashbourn, Julian (2020). *Audio Technology, Music, and Media*. 1. Berkhamsted: Springer Cham.
- Audioengine (o.D.). *What Is Audio Compression (And Why Should You Care)?* Abgerufen am 31.08.2024. URL: <https://audioengine.com/explore/what-is-audio-compression-and-why-should-you-care/>.
- Auersignal (o. D.). *Signaltöne*. Abgerufen am 19.08.2024. URL: <https://www.auersignal.com/de/technische-informationen/akustische-signalgerate/signaltone/>.
- Dr. Victor Ezquerro (Aug. 2020). *The Difference Between Acoustic, Electric and Digital Instruments*. Abgerufen am 19.08.2024. URL: <https://www.metromusicmakers.com/2020/08/the-difference-between-acoustic-electric-and-digital-instruments/>.
- e-teaching.org (o.D.). *WAV*. Abgerufen am 31.08.2024. URL: <https://www.e-teaching.org/materialien/glossar/wav>.
- Electronic Music Interactive (o.D.). *Digital Representation*. Abgerufen am 31.08.2024. URL: <https://pages.uoregon.edu/emi/8.php#:~:text=A%20digital%20representation%20of%20sound%20is%20a%20series%20of%20discrete,spaced%20measurements%20is%20a%20sample%20..>
- Karmasin, Matthias und Rainer Ribing (2019). *Die Gestaltung wissenschaftlicher Arbeiten. Ein Leitfaden für Facharbeit/VWA, Seminararbeiten, Bachelor-, Master-, Magister- und Diplomarbeiten sowie Dissertationen*. 10., überarbeitete und aktualisierte Auflage. Wien: facultas.
- Ken (Okt. 2021). *Manipulate Audio File in Python With 6 Powerful Tips*. Abgerufen am 31.08.2024. URL: <https://www.codeforests.com/2021/10/02/manipulate-audio-file-in-python/>.
- Lehrerinnenfortbildung Baden-Württemberg (o.D.). *Audioformate im Überblick*. Abgerufen am 31.08.2024. URL: https://lehrerfortbildung-bw.de/st_digital/medienwerkstatt/multimedia/audio/formate/.
- Librosa (o.D.). Abgerufen am 31.08.2024. URL: <https://github.com/librosa>.
- Meinard Müller and Vlora Arifi-Müller (o.D.). *Sheet Music Representations*. Abgerufen am 19.08.2024. URL: https://www.audiolabs-erlangen.de/resources/MIR/FMP/C1/C1S1_SheetMusic.html.

- Müller, Meinard (2021). *Fundamentals of Music Processing*. 2. Erlangen: Springer Cham.
- NRVision (o.D.). *Audioformate im Vergleich*. Abgerufen am 19.08.2024. URL: <https://www.nrwision.de/mitmachen/wissen/audioformate-vergleich>.
- NTi Audio (o.D.). *Fast Fourier Transformation FFT - Grundlagen*. Abgerufen am 31.08.2024. URL: <https://www.nti-audio.com/de/service/wissen/fast-fourier-transformation-fft>.
- Roxio (o.D.). *What is a WAV file?* Abgerufen am 31.08.2024. URL: <https://www.roxio.com/en/file-formats/wav-file/>.
- Stephen William (Okt. 2021). *Music Extraction*. Abgerufen am 31.08.2024. URL: <https://medium.com/@swilliam productions/music-extraction-7eb352d92bff>.

Anhang

- Eventuell Git-Repo verlinken