HOCHSCHULE KONSTANZ TECHNIK, WIRTSCHAFT UND GESTALTUNG
UNIVERSITY OF APPLIED SCIENCES

Signale, Systeme und Sensoren

Aufbau eines einfachen Spracherkenners

Th. Gnädig, F. Gendusa

Konstanz, 18. Dezember 2015

Zusammenfassung (Abstract)

Thema:	Aufbau eines einfachen Spracherkenners	
Autoren:	Th. Gnädig	thgnaedi@htwg-konstanz.de
	F. Gendusa	fagendus@htwg-konstanz.de
Betreuer:	Prof. Dr. Matthias O. Franz	mfranz@htwg-konstanz.de
	Jürgen Keppler	juergen.keppler@htwg-konstanz.de
	Martin Miller	martin.miller@htwg-konstanz.de

In diesem Versuch wird ein einfacher Spracherkenner aufgebaut. Insgesamt soll er vier Befehle voneinander unterscheiden können.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listingverzeichnis

1

Einleitung

In diesem Versuch soll ein einfacher Spracherkenner aufgebaut werden. Insgesamt soll er einen Wortschatz haben, der sich über vier Wörter erstreckt. Diese sind:

- Hoch
- Tief
- Links
- Rechts

So könnte der Spracherkenner beispielsweise für die Steuerung eines Staplers eingesetzt werden. Das Prinzip der Spracherkennung folgt nach dem Prinzp des Prototyp-Klassifikators. Weiter wird die Windowing-Medthode eingesetzt. Die Spracherkennung erfolgt in der Fourierdomäne. Das heißt es sollen die Spektren verglichen werden. Die Aufgabe teilt sich in zwei Versuche auf, so dass das Ziel Spracherkennung in zwei Schritte aufgeteilt wird.

- Fourieranalyse lang andauernder Signale
- Spracherkennung

In der ersten Teilaufgabe werden die Vorbereitungen auf den zweiten Versuch - die eigentliche Spracherkennung durchgeführt. Im Zweiten werden dann die im ersten teil gewonnenen Ergebnisse für die Spracherkennung angewandt. Der Spracherkenner wird in der Programmiersprache Python implementiert.

2

Fourieranalyse lang andauernder Signale

In diesem Versuch werden grundlegende Funktionen, die für Spracherkennung notwendig sind implementiert.

2.1 Fragestellung, Messprinzip, Aufbau, Messmittel

Zunächst wird eine Funktion implementiert, um eine Aufnahme von einem Mikrofon aufnehmen zu können. Dieses ist an einem Computer angeschlossen, welcher die Spracheingabe anschließend abgespeichert. Ein Bild einer Beispielhaften Aufnahme mit dem Wort 'Wurst' ist in den Messergebnissen zu sehen. Abbildung **??** Diese Aufnahme erstreckt sich über zwei Sekunden. Eine Aufnahmezeit von einer Sekunde würde für die geforderten Wörter ausreichen. Allerdings hat man so einen größeren zeitlichen Bereich, in welchem man das entsprechende Wort aufnehmen kann. Da sich der Beginn eines Wortes in der Aufnahme an unterschiedlichen Stellen befinden kann ist es nötig zu erkennen, wann das eigentliche Wort in der Aufnahme beginnt. Der Anfang des Wortes wird mit einem Trigger bestimmt. Das bedeutet, dass ein Teil der Aufnahme, von Aufnahmebeginn, bis das Signal einen gewissen Schwellenwert erreicht hat, abgeschnitten wird. So wird das Wort an den Aufnahmebeginn gesetzt. Weiter wird die Aufnahme auf eine Sekunde getrimmt. Für den Fall, dass das Signal nach Triggerung, kürzer als eine Sekunde ist wird es mit Nullen aufgefüllt. Ist das Signal noch länger als eine Sekunde wird es auf eine Sekunde beschnitten. Zudem wird das Signal nochmals Rückwarts durchlaufen und auf Null gesetzt, bis ein gewisser Schwellenwert überschritten wird. Dieser 'rükwerts Trigger' dient zur beseitigung des anschließenden Rauschens. Mit der Triggerung wird Sichergestellt, dass die später zu vergleichenden Worte auch zugleich beginnen. Von diesem Signal wird das Amplitudenspektrum errechnet. Die Darstellung des Amplitudenspektrums ist in Abbildung **??** Bei lang andauernden Signalen könnte

man prinzipiell das Spektrum sehr genau messen. Oftmals möchte man dies aber nicht und zerlegt das Signal in viele Abschnitte. Damit erreicht man eine höhere zeitliche Lokalität. Allerdings hat man auch ein breiteres Frequenzband, da man nach der Frequenz-Zeit Komplementarität Zeit und Frequenz nicht gleichzeitig beliebig genau messen kann. [**?** , S.7] Ausserdem steigt auch die Berechnungsdauer für die numerische Fouriertransformation mit $O(N \log N)$. Auch von diesem Aspekt ist eine Zerlegung in einzelne Fenster vorteilhaft. [**?** , S.16] Diese Methode, das sogenannte Windowing wird auch in diesem Versuch eingesetzt. Doch dieser Naive Ansatz muss noch erweitert werden, damit er auch wirklich praxistauglich ist. Unterteilt man das Fenster hart in verschiedene Signalabschnitte holt man sich starke Sprünge ins Signal(Unstetigkeiten). Diese schnellen Sprünge bringen hohe Frequenzen mit in das Signal hinein, da sich dieses an diesen Stellen schnell verändert. Hier gilt der Grundsatz, dass sich ein Signal nicht schneller verändern kann als sein Sinusanteil mit höchster Frequenz. Diese Sprünge an den Rändern des Fensters werden als schnelle Signaländerungen interpretiert, die im ursprünglichen zusammenhängenden Signal nicht vorhanden waren. Da dieser Effekt fatale Folgen für das Spektrum des Signals hätte kann nicht einfach so naiv vorgegangen werden. Um diesen Effekt zu kompensieren wird der Ansatz des Windowing erweitert. Diese Erweiterung sieht eine Multiplikation des Signals mit einer Fensterfunktion vor. Diese ist eine gerade Funktion, die in ihrer Symmetrieachse den Wert eins hat und an ihren enden links und rechts gegen Null konvergiert. Diese Eigenschaften treffen auf die Gaussfunktion zu, die in dieser Aufgabe als Fensterfunktion benutzt wird. Die Fensterbreite der Gaussfunktin beträgt hier einen Wert von vier Standartabweichungen. Auch hier spielt die Unschärferelation mit hinein. Es gilt je breiter die Fensterfunktion, desto größer die Frequenzauflösung und umgekehrt. [**?** , S.21] Windowing. Um nicht wertvolle Signalinformationen an den Rändern der Fensterfunktion zu verlieren müssen sich die Fenster zu 50 % überlappen. Die Fensterbreite ist in dieser Aufgabe 512 Samples. Nun kann in allen Fenstern lokal die Fouriertransformation durchgeführt werden. Aus den einzelnen Spektren wird das Arithmetische Mittel bestimmt. Welches anschließend als Spektrum des Signals angesehen.

2.2 Messwerte

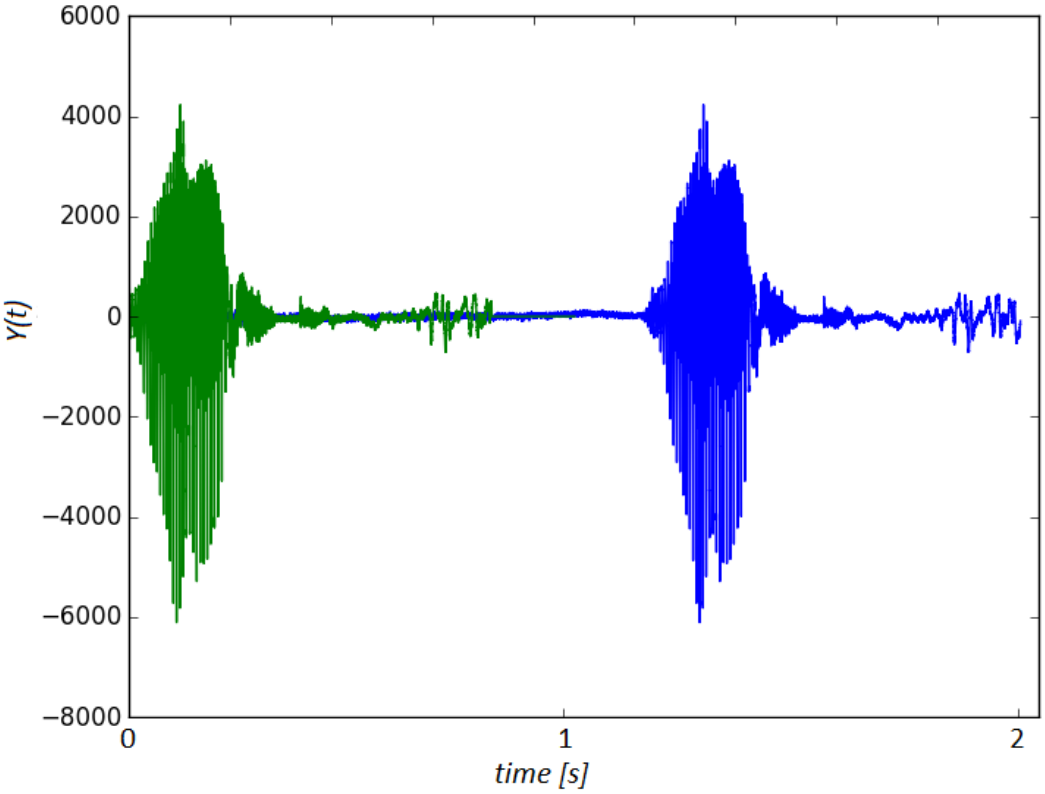


Abbildung 2.1: Das Wort ’Wurst’ in der Zeitdomäne.

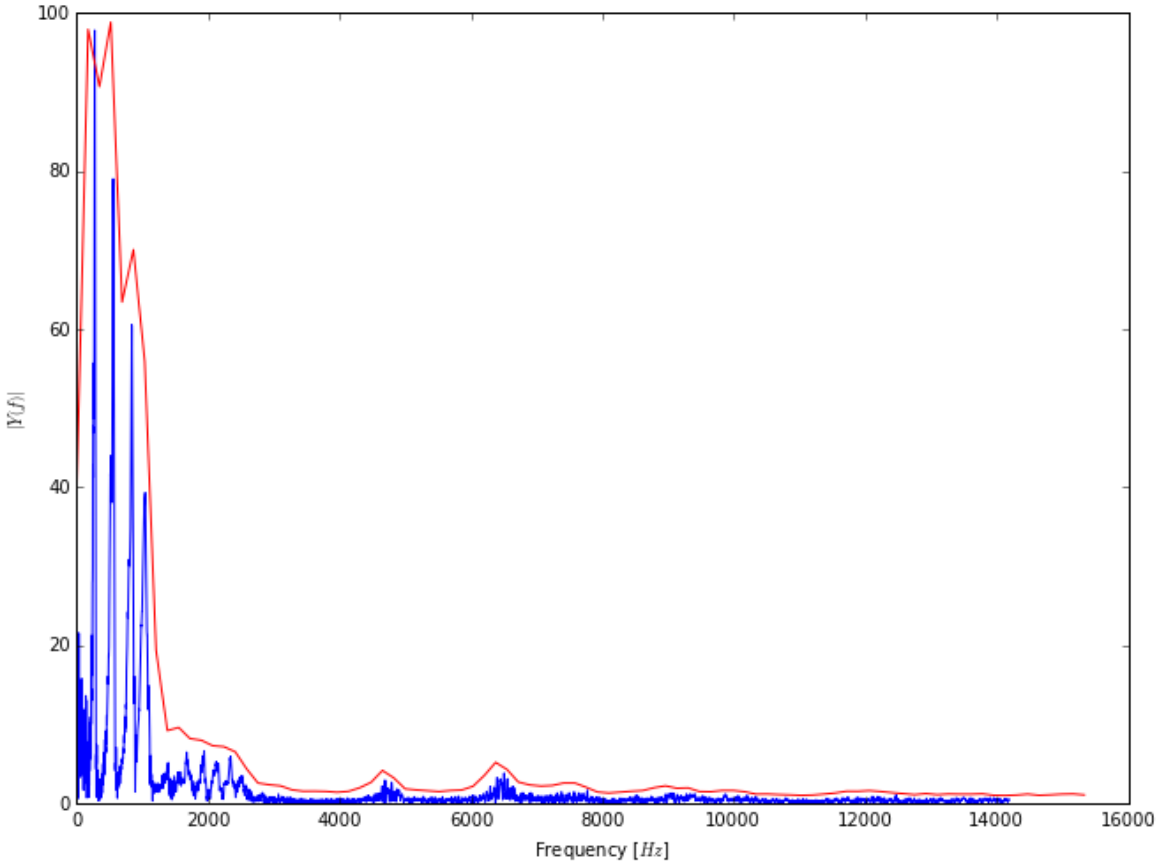


Abbildung 2.2: Das Wort 'Wurst' in der Frequenzdomäne

2.3 Auswertung und Interpretation

In Abbildung **??** ist ein Beispielhaftes Signal in der Zeitdomäne zu sehen. Der blaue Signalverlauf ist das Signal ohne Triggerung. Der grüne Graph ergibt sich aus dem blauen Signal nach Triggerung und Zuschnitt auf eine Sekunde. Die X-Achse stellt wie zu erwarten die Zeit in Sekunden dar. Die Einheit des Wertebereichs ist schwierig zu deuten, da den Versuchsdurchführenden nicht bekannt ist, was für Werte die Soundkarte zurückliefert. Das Ergebnis entspricht somit den Erwartungen. Für dieses Beispielhafte Signal ist in Abbildung **??** das Spektrum zu sehen. Die blauen Linien sind das Spektrum ohne Windowing. Die rote Linie ist das Spektrum mit Windowing. Bei diesem gilt selbiges für den Wertebereich, wie bei dem Signal im Zeitbereich. Die in diesem Versuch implementierten Funktionen werden im folgenden für die Implementierung des Spracherkenners benutzt.

3

Spracherkennung

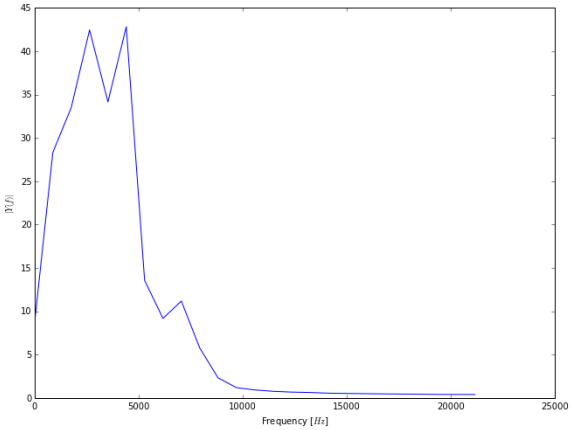
Dieser Abschnitt befasst sich mit der eigentlichen Spracherkennung und greift auf die Ergebnisse der vorigen Teilaufgabe zurück.

3.1 Fragestellung, Messprinzip, Aufbau, Messmittel

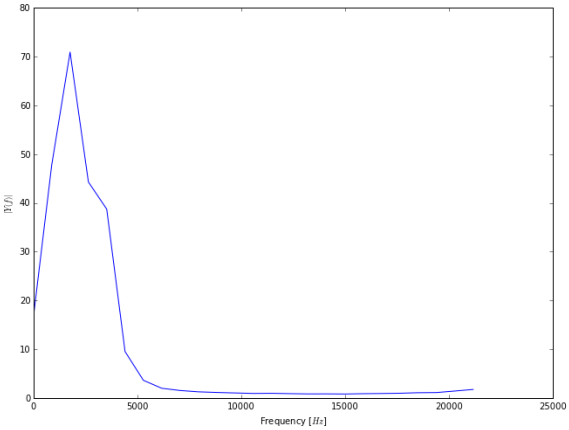
Für den Aufbau des Spracherkenners wird auf das in der Vorlesung behandelte Prinzip des Prototypenklassifikators zurückgegriffen. [**?** , S.24] Gemäß dieser Mustererkennung wurde für die Wörter *Hoch, Tief, Links, Rechts* jeweils ein Referenzspektrum gebildet. Diese Spektren werden in der Mustererkennung mit dem jeweiligen Eingabewort verglichen. Dieser Vergleich wird mit dem Verfahren der Kovarianz durchgeführt. Im Gegensatz zur Bestimmung der Ähnlichkeit mit Korrelation wird bei der Kovarianz zunächst bei jedem Signal der Mittelwert des selbigen abgezogen. Auf diese Weise können auch unterschiedlich große Signale, die den gleichen Signalverlauf haben als übereinstimmend erkannt werden. Praktisch wird dadurch die Möglichkeit gegeben ein Wort laut auszusprechen oder auch leise. Beide Male würde das Wort erkannt werden. Zum Vergleich der Signale findet der Korrelatioinskoeffizient nach Bravais-Pearson Einsatz, um die Spektren zu vergleichen. Der Korrelationskoeffizient ergibt sich aus dem Quotienten der Kovarianz beider Signale und dem Produkt der einzelnen Standartabweichungen. Der Korrelatinskoeffizient nach Bravais-Pearson hat somit einen Wertebereich von -1.0 bis 1.0. Werte nahe bei 1.0 bedeuten eine hohe Ähnlichkeit. Handelt es sich bei dem Ergebnis um die Zahl null ist keine Ähnlichkeit vorhanden. Bekommt man als Resultat eine Zahl nahe an -1 ist diese als Änti-Ähnlichkeitßu deuten.[**?** , S.27] Da ein Mensch kein technisches System ist, ist nicht gewährleistet, dass ein eben ausgesprochenes Wort im nächsten Moment exakt gleich klingt wenn es nochmals ausgesprochen wird. Aufgrund dieser Tatsache wird das Referenzspektrum aus dem Mittel der

Spektren von Fünf verschiedenen Aufnahmen gebildet. Für die Messung des Refereznspektrums muss darf der Sprecher nicht wechseln. Sind die Referenzspektren erstellt kann nun nach der Vorgestellten Methode der Kovarianz ein Vergleich stattfinden. Das Signalpaar, dass den größten Korrelationskoeffizienten hat, welcher zusätzlich noch größer als 0.9 sein muss wird als übereinstimmend betrachtet. Die Zahl 0.9 wurde experimentell ermittelt. Es wird also nicht einfach das Wort genommen, dass den größten Korrelationskoeffizienten hat. Würde man ein beliebiges Wort, für welches es kein Referenzspektrum gibt, in den Spracherkenner eingeben, wurde dieser sich für das Wort mit dem ähnlichsten Spektrum entscheiden, selbst wenn die Spektren sehr verschieden sind. Zum Testen des Spracherkenners werden Zwei Testdatensätze mit jeweils Fünf Aufnahmen für jedes Wort, das der Spracherkenner beherrscht aufgenommen. Bei dem ersten Testdatensatz spricht der Sprecher, der auch in den Aufnahmen für das Referenzspektrum gesprochen hat. Im zweiten Satz spricht ein anderer Sprecher. Es ergeben sich 40 Sprachaufnahmen, die automatisiert in den Spracherkenner geladen werden. Das Ergebnis wird untersucht, ob es dem Erwarteten entspricht, um so die Zuverlässigkeit des Spracherkenners zu testen. Der Spracherkenner selbst bekommt diese Testaufnahmen im Zeitbereich und muss zuerst mit den aus dem ersten Aufgabenteil entwickelten Methoden eine Fourieranalyse durchführen. Diese Aufnahmen wurden bei der Aufnahme auch schon getriggert.

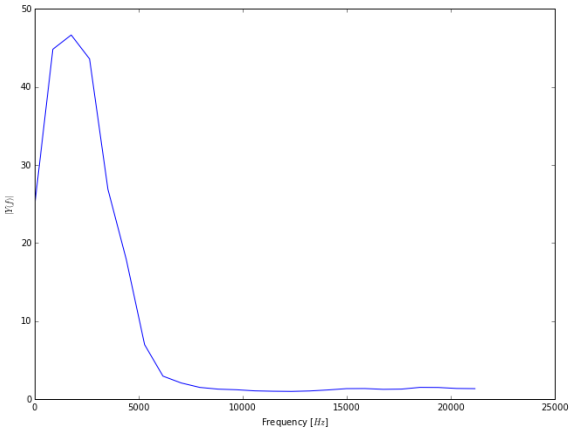
3.2 Messwerte



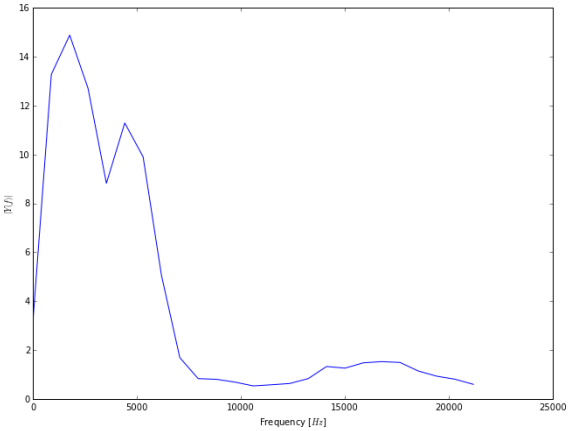
(a) Hoch



(b) Tief



(a) Links



(b) Rechts

Abbildung 3.2: Referenzspektren der einzelnen Wörter

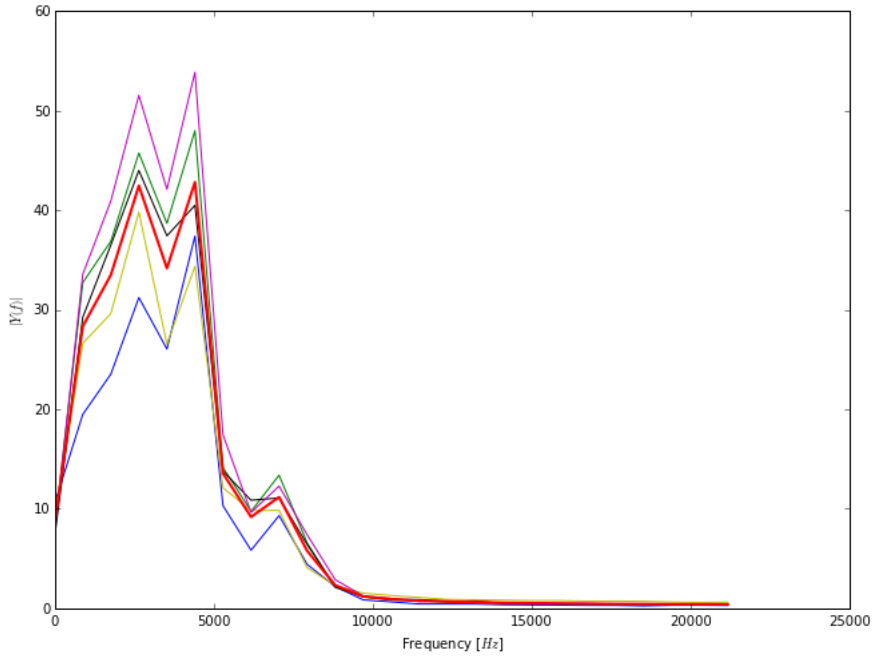


Abbildung 3.3: Mittelwert eines Referenzspektrums mit einzelnen Spektren

3.3 Auswertung und Interpretation

In Abbildung 3.2 sind die Referenzspektren zu den einzelnen Worten zu sehen. Auf der X-Achse ist die Frequenz aufgetragen und auf der Y-Achse die Amplitude. Die Frequenzen mit den höchsten Amplituden halten sich in einem Bereich von 0 - $\approx 10kHz$ auf. Dieses Ergebnis passt in das Frequenzband der Menschlichen Stimme hinein. [**?**] Bei den Wörtern Rechts und Links scheinen mit die Größten Frequenzen vorhanden zu sein. Diese übersteigen das Frequenzband der menschlichen Stimme und werden daher auf den Zischlaut s zurückgeführt, der in beiden Worten Links bzw. Rechts vorhanden ist. Abbildung 3.3 sind die Spektren der einzelnen Messungen für das Wort 'Hoch', verschiedenfarbig zu sehen. Die etwas dickere rote Linie ist der Mittelwert über alle aufgenommenen Spektren des Signals und wird als Referenzspektrum für das Wort Hoch benutzt. Es zeichnet sich eine Charakteristik für das Wort 'Hoch' ab, welche sich im Mittelwert widerspiegelt.

Insgesamt wurden 34 von 40 Wörtern richtig erkannt. Prozentual bedeutet das, dass der Spracherkenner eine Trefferquote von 85% hat. Bei den Testaufnahmen des Referenzsprechers gab es eine Trefferquote von 100%. Die Aufnahmen des anderen Sprechers wurden zu 70% korrekt erkannt. Sechs von Zwanzig Aufnahmen sind von diesem Testdatensatz nicht korrekt erkannt worden. Dieses Ergebnis ist erfreulich. Es entspricht den Erwartungen, dass der Testdatensatz des Sprechers, der auch die Referenzaufnahmen gesprochen hat besser ko-

rellieren, da der Spracherkenner quasi auf seine Stimme getrimmt wurde. Nicht jeder Mensch hat die gleiche Stimmtonlage, deshalb wurden auch weniger Wörter des zweiten Testsprechers erkannt. Allerdings sind mehr als die Hälfte der Wörter, welche der Sprecher B sprach erkannt wurden. Auch dies ist ein positives Resultat. Der Spracherkenner scheint somit gelungen zu sein.

Anhang

A.1 Quellcode

A.1.1 Quellcode Versuch 1

```
import pyaudio
import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as wim
import scipy.stats

FORMAT = pyaudio.paInt16
SAMPLEFREQ = 44100
FRAMESIZE = 1024
NOFFRAMES = 220
TRIGGER = 400

def trigger(signal):
    for i in range(0,len(signal)):
        if signal[i] > TRIGGER:
            signal = signal[i:]
            break
    restlength = min(len(signal),44100)
    zeros = np.array(np.zeros(44100))
    for i in range(0, restlength):
        zeros[i] = signal[i]
    for i in range(0,44100):
        if zeros[44099-i] > TRIGGER/2:
            break
        zeros[44099-i] = 0
    return zeros

def record():
```

```
    p = pyaudio.PyAudio()
    print("running")
    stream = p.open(format=FORMAT, channels=1, rate=SAMPLEFREQ, input=True)
    data = stream.read(SAMPLEFREQ*2)
    decoded = np.fromstring(data, 'Int16')
    stream.stop_stream()
    stream.close()
    p.terminate()
    print("done")
    decoded = trigger(decoded)
    plt.plot(decoded)
    plt.show()

    file = "fab.csv"
    np.savetxt(file,decoded,delimiter=',')
    return decoded

def fft_window(signal):
    gauss = wim.gaussian(512,std=512/4)
    fourier = np.zeros(256)
    tmp = np.zeros(512)
    for i in range(0,44100 - 512, 256):
        for a in range(0,512):
            tmp[a] = gauss[a] * signal[i + a]
        fourier = fourier + np.abs(np.fft.fft(tmp, axis=0)[:256])
    fourier = (fourier / 171)/450
    return fourier

def fft_std(data, filename=''):
    fft = np.abs(np.fft.fft(data, axis=0))
    plt.figure(figsize=(800/75, 600/75), dpi=75)
    plt.plot(fft[:5000]/len(fft))
    plt.autoscale(enable=True,axis = "x",tight = True)
    plt.xlabel('Frequency [$Hz$]')
    plt.ylabel('$\\left| Y \\left( f \\right ) \\right|$')
    if filename is not '':
        plt.savefig(filename+'.png', dpi=75)
        plt.close()
    plt.show()
    return

def ref_spektren(filename):
```

```
71      spektrum = np.zeros(256)
72      for id in range(0,5):
73          data = np.genfromtxt("{}_{}.csv".format(filename, id), delimiter=',')
74          data = fft_window(data) #spektrum erzeugt
75          spektrum = spektrum + data;
76      spektrum = spektrum / 5
77      return spektrum
78
79  def get_freq(M):
80      dt = 1 / SAMPLEFREQ
81      freq = np.zeros(M)
82      for i in range(0,M):
83          freq[i] = i / (M*dt)
84      return freq
85
86  def save_spekt(filename):
87      plt.figure(figsize=(800/75, 600/75), dpi=75)
88      ref = ref_spektren(filename)
89      plt.xlabel('Frequency [$Hz$]')
90      plt.ylabel('$\\left| Y \\left( f \\right ) \\right|$')
91      plt.plot(get_freq(50)[:25],ref[:25],c='b',lw=1)
92      plt.show()
93      np.savetxt(filename,ref,delimiter=',')
94
95  def korellation(spekA, spekB):
96      return scipy.stats.pearsonr(spekA, spekB)
97
98  def spracherkenner(wort):
99      hoch = np.genfromtxt("Aufnahmen_Referenz_Hoch", delimiter=',')
100     tief = np.genfromtxt("Aufnahmen_Referenz_Tief", delimiter=',')
101     links = np.genfromtxt("Aufnahmen_Referenz_Links", delimiter=',')
102     rechts = np.genfromtxt("Aufnahmen_Referenz_Rechts", delimiter=',')
103     wortschatz = { "Hoch" : hoch, "Tief" : tief, "Links" : links, "Rechts": rechts}
104
105     wort = fft_window(wort)
106     max = 0;
107     finalkey = "nicht gefunden"
108
109     for key in wortschatz:
110         korrel = korellation(wort,wortschatz[key])
111         if(korrel[0] > max):
112             max = korrel[0]
```

```
            finalkey = key


    if(max < 0.9):
        finalkey = "nicht gefunden"
    else:
        print("Wort: {}, Korrellation: {}".format(finalkey,max))
    return finalkey

def all_data():
    number = 0
    success = 0
    for sprecher in ["A","B"]:
        for wort in ["Hoch","Tief","Links","Rechts"]:
            for i in [0,1,2,3,4]:
                number = number + 1
                print("Datei : Aufnahmen_Test_{}_{}_{}.csv:".format(sprecher, wort, i))
                finalkey = spracherkenner(np.genfromtxt("Aufnahmen_Test_{}_{}_{}.csv".format(sprecher, wort, i), delimiter=','))
                if(finalkey == wort):
                    print("Happy")
                    success = success+1

    print("Erfolgreich festgestellt sind {} von {} worten, {}%".format(success,number,(success / number) * 100))
    return
all_data()
```