YAZILIM MÜHENDİSLİĞİ SİNYALLER ve SİSTEMLER ÖDEVİ

Adı Soyadı:Enes Küccüksolak

No: 202413211808

---

1. **3500 Hz Yüksek Geçiren Filtre Uygulaması**

Bu kısımda, 3500 Hz yüksek geçiren bir filtre kullanarak ses sinyali üzerinde işlem yaptık. Bu filtre, 3500 Hz'nin altındaki düşük frekansları bastırıyor ve sadece 3500 Hz'nin üstündeki yüksek frekansları geçiriyor. Aşağıda filtreleme işlemiyle ilgili bilgiler yer alıyor.

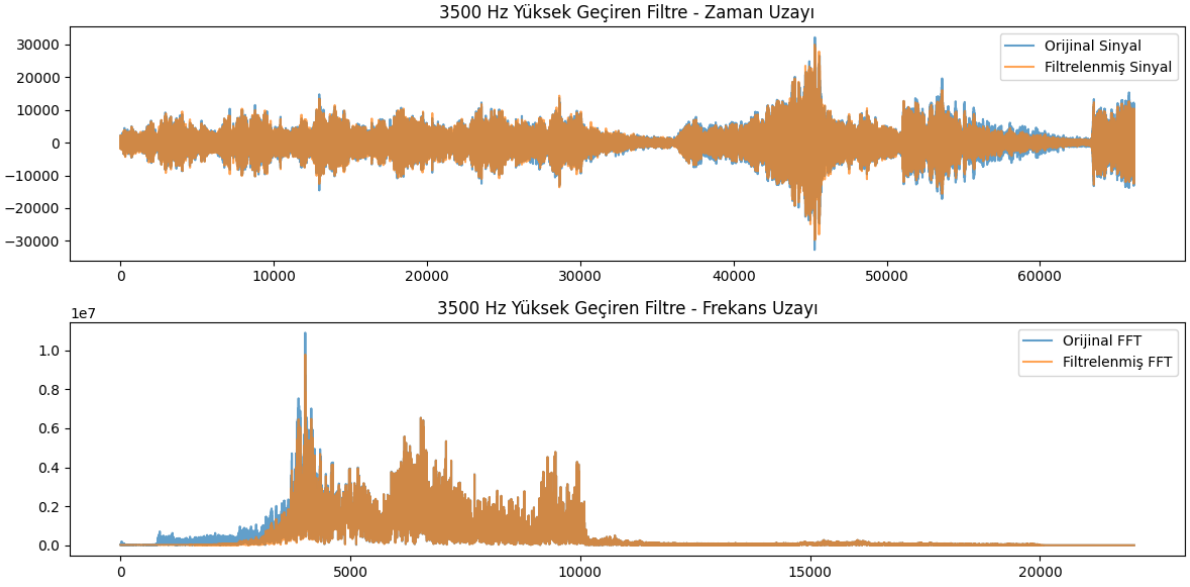
Filtreleme Sistemine Ait Detaylar

1. **Örnekleme Frekansı:**
   Sinyalin örnekleme frekansı 44100 Hz. Bu, sinyalin saniyede 44100 kez örneklendiği anlamına geliyor. Genelde ses dosyaları için standart bir değer.
2. **Filtrenin Tipi:**
   Kullanılan filtre yüksek geçiren bir filtre. Bu filtre, düşük frekansları azaltarak sadece yüksek frekansların sinyalde kalmasını sağlıyor.
3. **Kesim Frekansı:**
   Filtrenin kesim frekansı 3500 Hz olarak belirlendi. Bu da, 3500 Hz'nin üstündeki frekansların korunacağı anlamına geliyor.
4. **Filtre Mertebesi:**
   5. dereceden bir Butterworth filtresi kullandık. Bu filtre, geçiş bölgesinde düzgün bir yanıt verdiği için tercih edildi.

Filtreleme İşlemi

Filtreleme işlemi sırasında iki farklı grafikten faydalandık:

- **Zaman Alanı (Time Domain):**
  - İlk olarak, orijinal sinyalin dalga formunu inceledik. Bu, sinyalin zaman içinde nasıl değiştiğini gösteriyor. Filtreleme işleminden sonra, sinyalin düşük frekanslı kısımları yok oldu ve sadece yüksek frekanslı kısımlar kaldı.
- **Frekans Alanı (Frequency Domain):**
  - Orijinal sinyalde, geniş bir frekans aralığına sahip olduğumuzu gördük. Filtreleme işleminden sonra, sadece 3500 Hz'nin üstündeki frekansların sinyalde kaldığını fark ettik.


3500 Hz Yüksek Geçiren Filtre - Zaman Uzayı
Orijinal Sinyal
Filtrelenmiş Sinyal
30000
20000
10000
0
−10000
−20000
−30000
0
10000
20000
30000
40000
50000
60000

3500 Hz Yüksek Geçiren Filtre - Frekans Uzayı
1e7
Orijinal FFT
Filtrelenmiş FFT
1.0
0.8
0.6
0.4
0.2
0.0
0
5000
10000
15000
20000

2. 2500 Hz Alçak Geçiren Filtre Uygulaması

Bu bölümde, 2500 Hz alçak geçiren bir filtre kullanılarak ses sinyali üzerinde işlem yapıldı. Bu filtre, 2500 Hz'nin altındaki düşük frekansların geçmesine izin verirken, 2500 Hz'nin üstündeki yüksek frekansları bastırıyor.

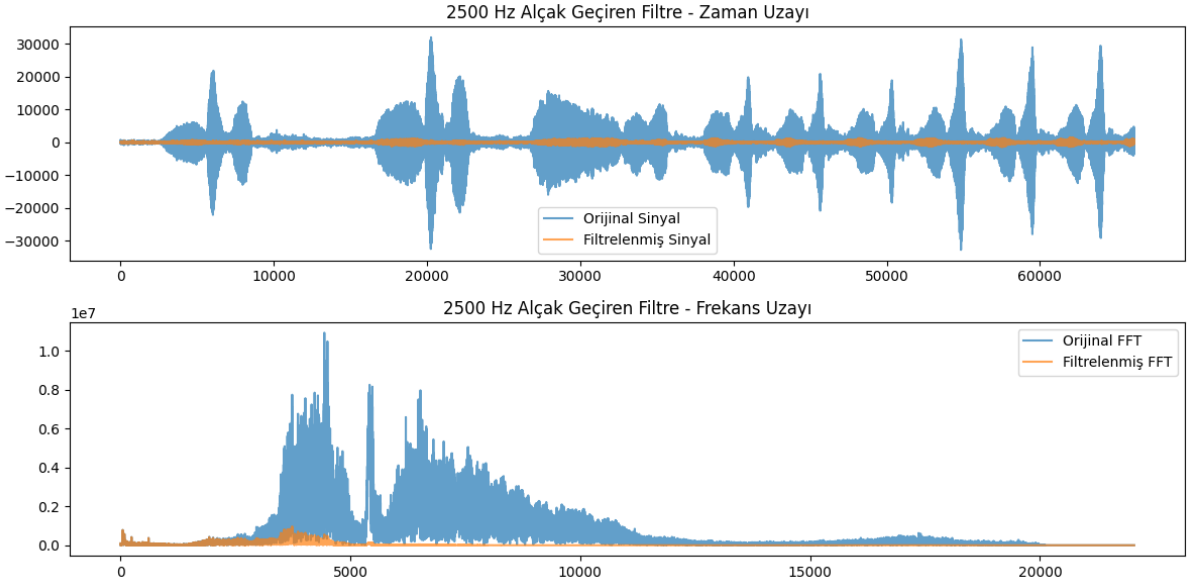
Filtreleme Sistemine Ait Detaylar

1. **Örnekleme Frekansı:**
    a. Bu sinyalin örnekleme frekansı 44100 Hz. Bu, saniyede 44100 tane örnek alındığı anlamına geliyor ve ses sinyali için standart bir değer.
2. **Filtrenin Tipi:**
    a. Kullandığımız filtre, alçak geçiren bir filtre. Bu filtre, düşük frekansları koruyup yüksek frekansları bastırır.
3. **Kesim Frekansı:**
    a. Filtrenin kesim frekansı 2500 Hz. Bu, 2500 Hz'nin altındaki frekansların sinyalde kalacağı, üstündekilerin azaltılacağı anlamına geliyor.
4. **Filtre Mertebesi:**
    a. 5. dereceden bir Butterworth filtresi kullandık. Butterworth filtreleri, geçiş bölgesinde pürüzsüz bir yanıt verdiği için tercih edilir.

Filtreleme İşlemi

Bu filtreleme işleminde sinyalin hem zaman hem de frekans alanındaki etkilerini inceledik:

- **Zaman Alanı (Time Domain):**
    - İlk grafikte orijinal sinyalin dalga formunu inceledik. Bu, sinyalin zaman içindeki davranışını gösteriyor. Filtre uygulandıktan sonra, yüksek frekanslı titreşimlerin kaybolduğunu ve dalga formunun daha pürüzsüz hale geldiğini gördük.
- **Frekans Alanı (Frequency Domain):**
    - Fourier dönüşümüyle orijinal sinyalde geniş bir frekans dağılımı olduğunu gözlemledik. Ancak filtreleme işleminden sonra, sadece 2500 Hz'nin altındaki frekanslar kaldı ve üstündeki frekans bileşenleri büyük ölçüde azaldı.


2500 Hz Alçak Geçiren Filtre - Zaman Uzayı
30000
20000
10000
0
−10000
−20000
−30000
0
10000
20000
30000
40000
50000
60000
Orijinal Sinyal
Filtrelenmiş Sinyal

2500 Hz Alçak Geçiren Filtre - Frekans Uzayı
1e7
1.0
0.8
0.6
0.4
0.2
0.0
0
5000
10000
15000
20000
Orijinal FFT
Filtrelenmiş FFT

3. 2000-3000 Hz Bant Geçiren Filtre Uygulaması

Bu bölümde, 2000-3000 Hz arasında çalışan bir bant geçiren filtre kullanılarak ses sinyali üzerinde işlem yapıldı. Bu filtre, sadece 2000 Hz ile 3000 Hz arasındaki frekansları geçiriyor ve bu aralığın dışındaki frekansları bastırıyor.

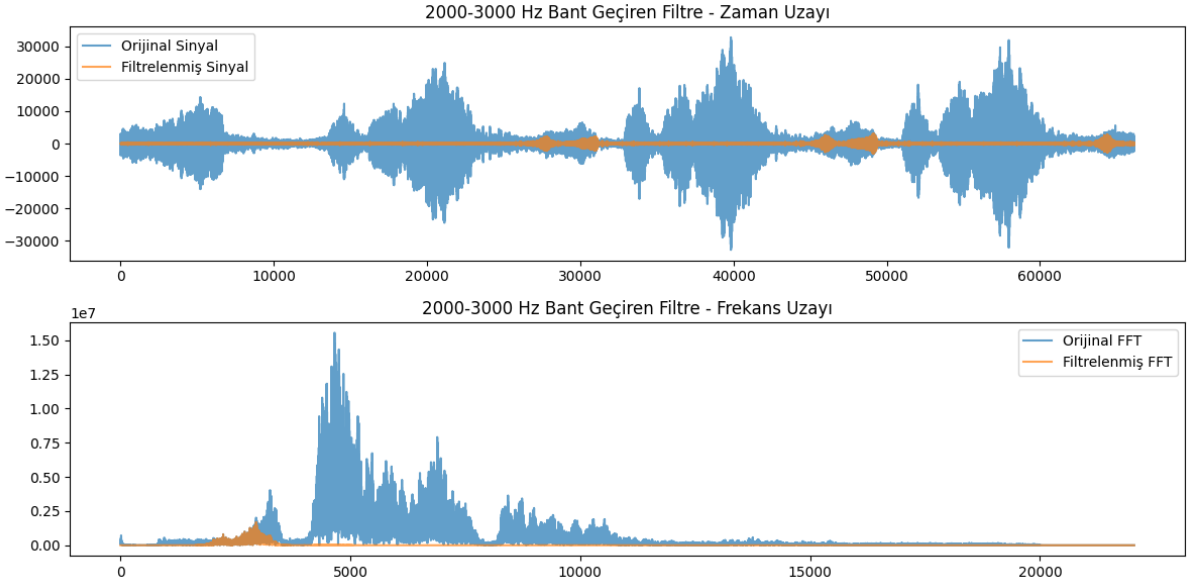
Filtreleme Sistemine Ait Detaylar

1. **Örnekleme Frekansı:**
    a. Ses sinyalinin örnekleme frekansı 44100 Hz. Bu, sinyalin saniyede 44100 kez örneklendiği anlamına gelir ve kaliteli bir ses işleme standardıdır.
2. **Filtrenin Tipi:**
    a. Bu filtre bir **bant geçiren filtre**dir. Bu tür filtreler, belirtilen frekans aralığını geçirirken, bu aralığın altındaki ve üstündeki frekansları baskılar.
3. **Kesim Frekansları:**
    a. Filtre, 2000 Hz ile 3000 Hz arasındaki frekans bileşenlerini geçirecek şekilde ayarlanmıştır. Bu aralık dışındaki frekanslar filtre tarafından baskılanır.
4. **Filtre Mertebesi:**
    a. 5. dereceden bir Butterworth filtresi kullandık. Bu filtre, geçiş bölgesinde düzgün bir yanıt sağladığı için sıkça tercih edilir.

Filtreleme İşlemi

Filtreleme sırasında sinyalin hem zaman hem de frekans alanındaki etkilerini inceledik:

- **Zaman Alanı (Time Domain):**
    - İlk olarak, orijinal sinyalin zaman eksenindeki dalga formunu inceledik. Filtreleme sonrası sinyalin sadece hedeflenen frekans aralığını içermesi nedeniyle dalga formu gözle görülür şekilde değişti.
- **Frekans Alanı (Frequency Domain):**
    - Fourier dönüşümüyle orijinal sinyalde tüm frekans bileşenlerinin mevcut olduğunu gördük. Ancak filtreleme sonrası sadece 2000-3000 Hz aralığında kalan frekans bileşenlerinin korunduğu görüldü.


2000-3000 Hz Bant Geçiren Filtre - Zaman Uzayı
Orijinal Sinyal
Filtrelenmiş Sinyal
30000
20000
10000
0
−10000
−20000
−30000
0
10000
20000
30000
40000
50000
60000

2000-3000 Hz Bant Geçiren Filtre - Frekans Uzayı
1e7
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Orijinal FFT
Filtrelenmiş FFT
0
5000
10000
15000
20000

4. EKLER:

```
#3500 Hz Yüksek Geçiren Filtre Uygulaması
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import butter, sosfilt
from pydub import AudioSegment

# Ses dosyasını yükleme
file_path = r"C:\Users\sertv\Desktop\Python\3500_high.mp3"  # Yüklediğimiz
dosyanın doğru yolu
audio = AudioSegment.from_file(file_path)
samples = np.array(audio.get_array_of_samples())
sample_rate = 44100  # Örnekleme frekansı (varsayılan olarak 44.1 kHz
kullanılır)

# Yüksek Geçiren Filtre Tasarımı
high_cut = 3500  # Kesim frekansı
order = 5  # Filtre mertebesi
sos = butter(order, high_cut, btype='highpass', fs=sample_rate, output='sos')

# Filtreleme işlemi
filtered_signal = sosfilt(sos, samples)

# Frekans uzayı analizi
freqs = np.fft.rfftfreq(len(samples), d=1/sample_rate)
original_fft = np.abs(np.fft.rfft(samples))
filtered_fft = np.abs(np.fft.rfft(filtered_signal))

# Grafikler
plt.figure(figsize=(12, 6))

# Zaman ekseninde sinyal
plt.subplot(2, 1, 1)
plt.plot(samples, label="Orijinal Sinyal", alpha=0.7)
plt.plot(filtered_signal, label="Filtrelenmiş Sinyal", alpha=0.7)
plt.title("3500 Hz Yüksek Geçiren Filtre - Zaman Uzayı")
plt.legend()

# Frekans ekseninde sinyal
plt.subplot(2, 1, 2)
plt.plot(freqs, original_fft, label="Orijinal FFT", alpha=0.7)
plt.plot(freqs, filtered_fft, label="Filtrelenmiş FFT", alpha=0.7)
plt.title("3500 Hz Yüksek Geçiren Filtre - Frekans Uzayı")
plt.legend()

plt.tight_layout()
plt.show()
```

```python
#2500 Hz Alçak Geçiren Filtre Uygulaması
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import butter, sosfilt
from pydub import AudioSegment

# Ses dosyasını yükleme
file_path = r"C:\Users\sertv\Desktop\Python\2500_lowpass.mp3"  # Yüklediğiniz dosyanın doğru yolunu buraya yazın
audio = AudioSegment.from_file(file_path)
samples = np.array(audio.get_array_of_samples())
sample_rate = 44100  # Örnekleme frekansı (varsayılan olarak 44.1 kHz kullanılır)

# Alçak Geçiren Filtre Tasarımı
low_cut = 2500  # Kesim frekansı
order = 5  # Filtre mertebesi
sos = butter(order, low_cut, btype='lowpass', fs=sample_rate, output='sos')

# Filtreleme işlemi
filtered_signal = sosfilt(sos, samples)

# Frekans uzayı analizi
freqs = np.fft.rfftfreq(len(samples), d=1/sample_rate)
original_fft = np.abs(np.fft.rfft(samples))
filtered_fft = np.abs(np.fft.rfft(filtered_signal))

# Grafikler
plt.figure(figsize=(12, 6))

# Zaman ekseninde sinyal
plt.subplot(2, 1, 1)
plt.plot(samples, label="Orijinal Sinyal", alpha=0.7)
plt.plot(filtered_signal, label="Filtrelenmiş Sinyal", alpha=0.7)
plt.title("2500 Hz Alçak Geçiren Filtre - Zaman Uzayı")
plt.legend()

# Frekans ekseninde sinyal
plt.subplot(2, 1, 2)
plt.plot(freqs, original_fft, label="Orijinal FFT", alpha=0.7)
plt.plot(freqs, filtered_fft, label="Filtrelenmiş FFT", alpha=0.7)
plt.title("2500 Hz Alçak Geçiren Filtre - Frekans Uzayı")
plt.legend()

plt.tight_layout()
plt.show()
```

```python
#2000-3000 Hz Bant Geçiren Filtre Uygulaması
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import butter, sosfilt
from pydub import AudioSegment

# Dosya yükleme
file_path = r"C:\Users\sertv\Desktop\Python\2000_3000_bandpass.mp3"
audio = AudioSegment.from_file(file_path)
samples = np.array(audio.get_array_of_samples())
sample_rate = 44100  # Örnekleme frekansı

# Bant Geçiren Filtre Tasarımı
low_cut = 2000
high_cut = 3000
order = 5  # Filtre mertebesi
sos = butter(order, [low_cut, high_cut], btype='bandpass', fs=sample_rate,
output='sos')

# Filtreleme işlemi
filtered_signal = sosfilt(sos, samples)

# Frekans uzayı analizi
freqs = np.fft.rfftfreq(len(samples), d=1/sample_rate)
original_fft = np.abs(np.fft.rfft(samples))
filtered_fft = np.abs(np.fft.rfft(filtered_signal))

# Grafikler
plt.figure(figsize=(12, 6))

# Zaman ekseninde sinyal
plt.subplot(2, 1, 1)
plt.plot(samples, label="Orijinal Sinyal", alpha=0.7)
plt.plot(filtered_signal, label="Filtrelenmiş Sinyal", alpha=0.7)
plt.title("2000-3000 Hz Bant Geçiren Filtre - Zaman Uzayı")
plt.legend()

# Frekans ekseninde sinyal
plt.subplot(2, 1, 2)
plt.plot(freqs, original_fft, label="Orijinal FFT", alpha=0.7)
plt.plot(freqs, filtered_fft, label="Filtrelenmiş FFT", alpha=0.7)
plt.title("2000-3000 Hz Bant Geçiren Filtre - Frekans Uzayı")
plt.legend()

plt.tight_layout()
```

```
plt.show()
```