Παύλος Σαρίδης ΑΕΜ 3732

Αναφορά για τον Φιλτραρισμό Ηχητικού Σήματος

Εισαγωγή

Στην παρούσα αναφορά περιγράφεται ο φιλτραρισμός ενός ηχητικού σήματος μέσω της χρήσης πεπερασμένης κρουστικής απόκρισης φίλτρου χαμηλής συχνότητας.

Μεθοδολογία

Για την υλοποίηση του φιλτραρίσματος χρησιμοποιήθηκε ο γλώσσα προγραμματισμού Python. Ο κώδικας περιλαμβάνει τις εξής βασικές εργασίες:

1. **Φόρτωση Αρχείου WAV:** Το αρχείο ηχογράφησης φορτώνεται στο πρόγραμμα.
2. **Σχεδιασμός Φίλτρου Χαμηλής Συχνότητας:** Σχεδιασμός πεπερασμένης κρουστικής απόκρισης φίλτρου χαμηλής συχνότητας.
3. **Εφαρμογή Φίλτρου:** Εφαρμογή του φίλτρου στο ηχητικό σήμα.
4. **Σχεδίαση Συχνοτικής Απόκρισης:** Προαιρετική σχεδίαση και αποθήκευση της συχνοτικής απόκρισης του φίλτρου.

Κώδικας Python

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import wavfile

# Αλλαγή του Matplotlib backend σε Agg
import matplotlib
matplotlib.use('Agg')

def load_wav(filename):
    # Φόρτωση αρχείου WAV
    samplerate, data = wavfile.read(filename)
    return samplerate, data

def design_lowpass_filter(samplerate, cutoff=0.2, numtaps=300):
    # Σχεδιασμός απλού χαμηλοπερατού φίλτρου χρησιμοποιώντας το numpy
    nyquist = 0.5 * samplerate
    cutoff_normalized = cutoff / nyquist
    taps = np.sinc(2 * cutoff_normalized * np.arange(numtaps))
    taps *= np.blackman(numtaps)
    taps /= np.sum(taps)
    return taps

def apply_filter(data, taps):
    # Εφαρμογή του φίλτρου στο σήμα χρησιμοποιώντας το numpy
    filtered_data = np.convolve(data[:, 0], taps, mode='same')  #
Εξασφαλίζουμε ότι το data είναι 1D
    return filtered_data

def plot_frequency_response(taps, samplerate,
save_path='frequency_response.png'):
    # Σχεδίαση της συχνοτικής απόκρισης και αποθήκευση ως εικόνα
    w = np.fft.fftfreq(len(taps))
    response = np.fft.fft(taps)
    plt.plot(0.5 * samplerate * w, np.abs(response), 'b')
    plt.xlabel('Συχνότητα (Hz)')
    plt.ylabel('Κέρδος')
    plt.title('Συχνοτική Απόκριση')
    plt.ylim(-0.05, 1.05)
    plt.grid(True)
    plt.savefig(save_path)
    plt.close()  # Κλείσιμο του πλοτ για να αποφευχθεί η διαδραστική
προβολή

# Φόρτωση του αρχείου WAV
samplerate, data = load_wav('The Neighbourhood.wav')  #
Αντικαταστήστε με την πραγματική διαδρομή του αρχείου σας

# Σχεδιασμός και εφαρμογή του φίλτρου
taps = design_lowpass_filter(samplerate, cutoff=0.2)
filtered_data = apply_filter(data, taps)

# Προαιρετικά, σχεδίαση της συχνοτικής απόκρισης και αποθήκευση ως
εικόνα
plot_frequency_response(taps, samplerate,
save_path='frequency_response.png')
```

```
# Αποθήκευση του φιλτραρισμένου αρχείου
wavfile.write('filtered_output.wav', samplerate,
filtered_data.astype(data.dtype))
```