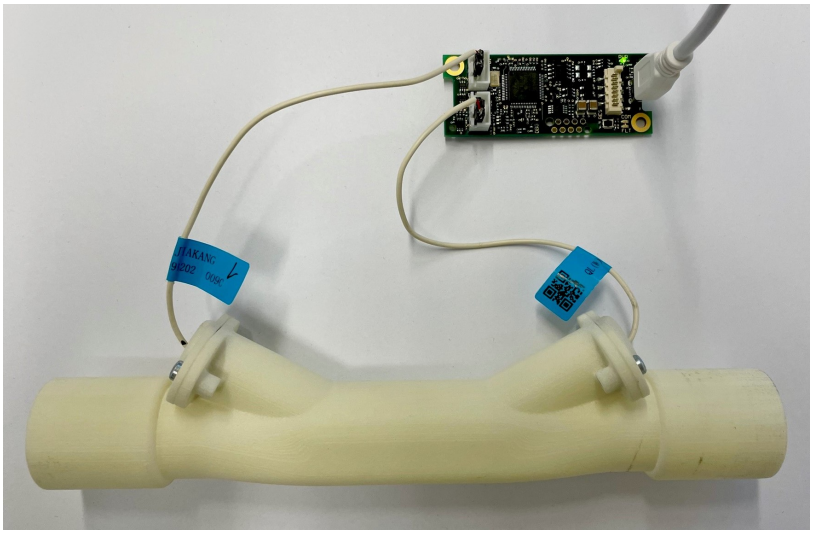
Tunable Bandpass Filter for Embedded Devices

Testatarbeit
Digitale Signalverarbeitung, HS 2021

Jürgen Wassner, Pascal Jund



Bearbeitung

Bearbeitung als 2er-Team oder Einzelarbeit

Abgabe

Alle Matlab-Files bis **Mo 13.12.21 12:00** als zip-Container `name1\_name2.zip` in ILIAS Briefkasten

Testat-Bedingung

1. Die Funktionen `bp_iir_bilin` and `bp_fir_win` sind funktionsfähig implementiert.
2. Das Live-Skript `bp_filt.mlx` erfüllt mindestens die Pflicht-Anforderungen und ist aussagekräftig kommentiert.

Nützliche Links

[1] Trapezregel und Bilineare Transformation

[2] Bilineare Transformation mit Prewarping

[3] Matlab: Bilinear transformation method for analog-to-digital filter conversion

Application

In an embedded volume-flow measurement system the transmitted ultra-sonic chirp is filtered at the receiver to limit signal bandwidth prior to further processing. For this a second-order IIR filter with bandpass characteristic is implemented on a low-cost microcontroller. Since the equipment shall be used for different applications requiring different frequency ranges, the center frequency and bandwidth of the bandpass filter shall be configurable.

The goal of this project is to implement and test the necessary code to calculate the filter coefficients for the second-order IIR filter in Matlab, when given the center frequency and bandwith of the filter. The code shall not use built-in Matlab functions such that a transfer to C-Code is easily possible. (The actual transfer to C-Code is not part of this project.)

Background Information

2nd-Order Bandpass Filter

The continuous-time transfer function of a second-order bandpass filter can be written as

$$H(p) \;=\; A \cdot \frac{\frac{B}{f_0 \cdot \omega_0} \cdot p}{1 + \frac{B}{f_0 \cdot \omega_0} \cdot p + \frac{1}{\omega_0{}^2} \cdot p^2} \tag{1}$$

where A denotes a frequency-independent amplification factor, $\omega_0 = 2\pi f_0$, and

$$Q \;=\; \frac{f_0}{B} \;=\; \frac{\sqrt{f_L \cdot f_U}}{f_U - f_L} \tag{2}$$

with

Q	...	quality factor (Güte) of the filter
B	...	filter bandwidth (frequency range above the -3 dB points)
f_L	...	lower pass band edge frequency (lower -3 dB point)
f_U	...	upper pass band edge frequency (upper -3 dB point)
f_0	...	center frequency of the filter (geometric mean of upper and lower frequency)

In this project (1) is the frequency response of the analog prototype filter which will be used for the design of 2nd-oder bandpass IIR filters with a discrete-time transfer function of the form

$$H(z) \;=\; K \cdot \frac{b_0 + b_1 \cdot z^{-1} + b_2 \cdot z^{-2}}{1 + a_1 \cdot z^{-1} + a_2 \cdot z^{-1}} \tag{3}$$

Bilinear Transform

A continuous-time transfer function $H(p)$ can be approximated by a time-discrete transfer function $H(z)$ by means of the following bilinear transform

$$p \;=\; \frac{2}{T_S} \cdot \frac{z-1}{z+1} \tag{4}$$

This transform maps the imaginary axes $j\omega$ of the p-plane onto the unit circle $z = e^{j\Omega}$ of the z-plane, such that a stable $H(p)$ will yield a stable $H(z)$, see [1]. However, mapping the indefinitely long imaginary axes onto the finite length unit circle will introduce distortion between the two frequency domains ω and Ω.

Bilinear Transform with Prewarping

The frequency distortion of the bilinear transform cannot be completely avoided but can be compensated such that the frequency response $H(\omega)$ of the continous -time system matches the time-discrete frequency response $H(\Omega)$ at a particular frequency point $\hat{f}$, i.e.

$$H(\omega)|_{\omega=2\pi\hat{f}} \;=\; H(\Omega)|_{\Omega=2\pi\hat{f}T_S} \tag{5}$$

To achieve this, the bilinear transform (4) is modified into

$$p \;=\; \frac{2\pi\hat{f}}{\tan(\frac{2\pi\hat{f}\cdot T_S}{2})}\cdot\frac{z-1}{z+1} \tag{6}$$

This is called bilinear transform with prewarping, which also yields a stable $H(z)$ if applied to a stable $H(p)$, see [2].
In the context of bandpass filter design it makes sense to use $\hat{f} = f_0$ such that the frequency response of the target IIR filter exactly matches the frequency response of the analog prototype filter at the center of the pass band.

Task Assignment

Mandatory Requirements

1. Implement a Matlab function `[b,a,K]=bp_iir_bilin(f_L,f_U,f_S,pw)` that returns the coefficient vectors of a second-order IIR filter `b=[b_0 b_1 b_2]` and `a=[1 a_1 a_2]` and a constant gain factor K based on the bilinear transform of the bandpass prototype filter (1) with center frequency f_0 and bandwidth B.
   Depending on parameter pw the bilinear transform is performed according to (4) or (6), i.e. without or with prewarping.

2. Implement a second Matlab function `[b]=bp_fir_win(f_L,f_U,N,f_S,win)` that returns the coefficient vector of an order N bandpass FIR filter using the window design method. Depending The function shall use a window function defined by parameter win (default = rectangular window).

3. Complete the given template of the Matlab live-script `tune_bp_filter.mlx` such that three figures are generated depending on parameters f_L, f_U, f_S, N_1, N_2 as follows:

   (a) The frequency response (magnitude [dB] and phase [rad]) of the analog prototype filter is displayed in figure #1.

   (b) The magnitude response [dB] of the following three systems are plotted together in figure #2:
   - analog prototype filter
   - IIR filter obtained from `bp_iir_bilin` without prewarping
   - IIR filter obtained from `bp_iir_bilin` with prewarping

   (c) The magnitude response [dB] of the following three systems are plotted together in figure #3:
   - IIR filter obtained from `bp_iir_bilin` with prewarping
   - FIR filter obtained from `bp_fir_win` with $N = N_1$ and rectangular window
   - FIR filter obtained from `bp_fir_win` with $N = N_2$ and rectangular window

4. Determine and set parameters N_1 and N_2 in `tune_bp_filter.mlx` such that

   N_1 the number of multiplications to be performed in the real-time implementation is the same for the second-order IIR filter and the FIR filter

   N_2 the magnitude response of the FIR filter is sufficiently close to the magnitude response of the IIR filter

Important Note:
The functions `bp_iir_bilin` and `bp_fir_win` to be implemented shall not use any built-in Matlab filter design function, but use operations only that are available in the standard C library.

Optional Requirements

1. Extend `tune_bp_filter.mlx` such that a fourth figure reproduces figure #3 but for a user-specified windowing function.

2. Determine and set an additional parameter N_3 in `tune_bp_filter.mlx` such that the magnitude response of the FIR filter designed with user-specified windowing function is sufficiently close to the magnitude response of the IIR filter. Compare N_3 with N_2. i.e. compare FIR filter order required to match IIR magnitude response when using rectangular and user-specified windowing function.