2015/2016

Digital Control in a Motional Feedback Audio System

EE3L11 — Bachelor Graduation Project
Thesis

Delft University of Technology

Project Team G's Controller team

Tijs Hol	4295528
Wouther Bons	4092716

Commissioned by:
Dr. Ir. G.M.Janssen

Friday 8th July, 2016

TU Delft Delft University of Technology

Challenge the future

Preface

This report is written as part of the Bachelor Graduation Project (BAP) for the Electrical Engineering bachelor at the TU Delft. The project group consists of six electrical engineering bachelor students, who are divided into subgroups of two and are each devoted to a subsystem of the total project.

The assignment was provided by Dr. Ir. G.J.M. Janssen out of interest in the digital implementation of the motional feedback system. Janssen provided further guidance during the project, for which we would like to thank him.

Other than Janssen we would like to thank the employees of the Tellegen Hall at the TU Delft for their help and the budget they provided. We also want to thank Ioan Lager for the coordination of the project. Lastly we want to thank the members of the jury for reviewing our work.

June 2016
Oscar de Groot, Jaco Salentijn, Ramon Overwater, Yann Rosema, Wouther Bons, Tijs Hol
TU Delft

Abstract

The quality of a loudspeaker system is largely determined by the distortion of the output signal, resulting from the non-ideal transfer of the loudspeaker and amplifier. This report proposes a method of reducing the distortion in the output signal by using a topology with active feedback. The feedback control unit, which is implemented by a microcontroller, approaches the problem with digital filters instead of analogue filters. This opens up a range of possibilities: fully automatic system identification, freely definable closed loop transfers and easy installing. Using MATLAB and Simulink, this report aims to make these possibilities a reality.

Contents

List of Variables

$a[n]$	Digitized acceleration ($a(t)$ after ADC)
$a(t)$	Acceleration of loudspeaker cone as measured by piezo-electric sensor
a_i	i^{th} zero of a system
A	Spectrum of acceleration of loudspeaker cone, as measured by piezo-electric sensor
α_n	n^{th} coefficient of a non-linear amplifier
$\mathbf{b_{lo,up}}$	Particle search space lower and upper bounds
$\boldsymbol{\beta}$	LLSO parameters
C	System identification function constant
c_v	Inertia weight
D	Transfer function of amplifier
E	Particle search space
$e[n]$	Error correction in input as calculated by controller
$e(t)$	Analogue error ($e[n]$ after DAC)
F_s	Sampling frequency
f	Frequency in Hertz
$f(x_i, \boldsymbol{\beta})$	Optimisation function
G	Transfer function of loudspeaker
$\mathbf{g}$	Swarm previous best position
H_0	Desired transfer of closed-loop system
H	Closed-loop transfer of system
I	PSO cycle limit
j	Unit imaginary number
K_0	Transfer function of base controller (i.e. no additional filters such as an LPF)
$k[n], k(t)$	Impulse response of controller
K	Transfer function of full controller (including additional filters such as an LPF)
$m[n]$	Measurement signal for system identification
M	LLSO fitting curve order
N	Data measurement length
ω	Frequency in radians per second
$\mathbf{p_i}$	Particle i previous best position
ϕ_p	Cognitive scaling parameter
ϕ_g	Social scaling parameter
p_i	i^{th} pole of a system
p_ϕ	System identification pole phase
r_p	Random cognitive factor
r_g	Random social factor
s	Laplace transform variable

S	Particle swarm size
T_s	Sampling Period
$u_0(t)$	Audio input signal straight from audio source
$u(t)$	Audio input signal that will be corrected by controller
U	Spectrum of input signal u(t)
$\mathbf{v_i}$	Particle i velocity
$w(t)$	Audio input signal corrected by controller and input for amplifier.
x_i	Known data points
$\mathbf{x_i}$	Particle i position
y_i	Measured data points
Y	Spectrum of output signal y(t)
z	$\mathcal{Z}$ transform variable

List of Abbreviations

ADC	Analogue-to-Digital Converter
DAC	Digital-to-Analogue Converter
DC	"Direct Current", electrical signal with zero frequency
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
IIR	Infinite Impulse Response
LLSO	Linear Least Squares Optimisation
LPF	Low-Pass Filter
LSO	Least Squares Optimisation
LTI	Linear and Time-Invariant
MFB	Motional feedback
PSO	Particle Swarm Optimisation
RMS	Root Mean Square
SNR	Signal-to-Noise Ratio
THD	Total Harmonic Distortion

Chapter 1

Introduction

1.1 Background[1]

1.1.1 History of motional feedback

In the year 1968 Philips released a paper on motional feedback (MFB),[3] a technique where electro-acoustic feedback is used to reduce the linear and non-linear distortion of a speaker system. Philips and various other manufacturers sold audio systems with the motional feedback system for about a decade. Unfortunately the costs of the analogue equipment necessary for the implementation did not make the system worthwhile in the long run and it disappeared from the market.

Recent development in the costs and capabilities of digital devices have provided an opportunity for bringing back the motional feedback system by designing a digital implementation. The advantages of the modern techniques can make motional feedback an affordable extension to modern day audio systems.

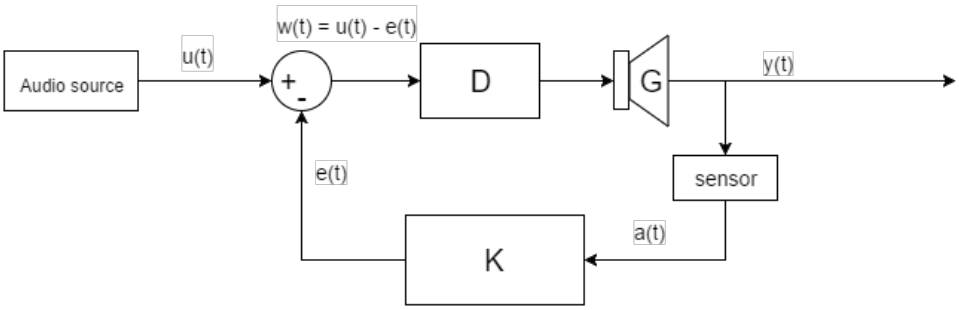
1.1.2 Principle of digital motional feedback

When playing sound over a loudspeaker, an electrical waveform resembling the sound is converted into air pressure waves which are sensed by human ears and perceived as sound by the brain. Because a loudspeaker is a non-ideal, physical system, this conversion from voltage to air pressure distorts the sound by for example damping out low frequencies (linear distortion) and adding harmonics that weren't present in the original waveform (non-linear distortion). Ideally, a loudspeaker would output the exact sound it receives at it's input, so finding a way to decrease this distortion is of interest.

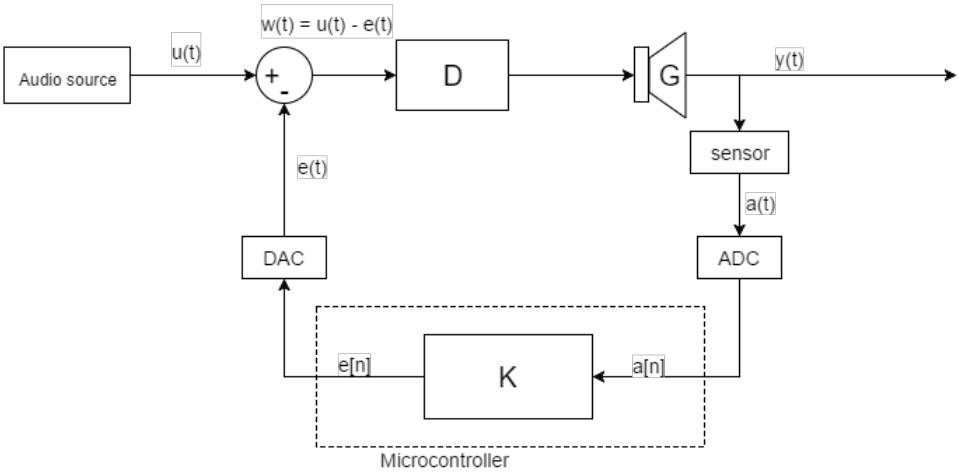
One way to do so is by using a motional feedback system, where the output of the speaker is measured by a sensor and used to calculate the error in the input signal. This error signal is then fed back to the input (fig. 1.1a) to correct it. To calculate this error signal from the measured output, an LTI filter called the *controller* is designed using Control System theory so that it corrects the input of the loudspeaker, thereby altering the output sound to more closely resemble the input waveform.

Digital MFB implements this controller on a digital platform. This requires sampling of the loudspeaker's measured output signal and reconstruction of the controller's output signal to an analogue voltage for comparison with the input signal (fig. 1.1b).

[1] Because it discusses the project in general, this section occurs in the theses of all three subgroups (see section 1.2).



(a) Controller implemented as analogue system.



(b) Controller implemented digitally.

Figure 1.1: Layout of a motional feedback system around a loudspeaker system with speaker G and amplifier D. Shown are the controller K, audio input signal $u(t)$, speaker output pressure $y(t)$, conus acceleration $a(t)$ and $a[n]$ and the error signal calculated by the controller $e(t)$ and $e[n]$.

1.2 Division into subsystems[2]

The design of the digital motional feedback device is split into three parts for the purpose of this project. The interfaces between these parts are defined as part of the Programme of Requirements (see chapter 2).

Digital signal processing
The first part is devoted to the digital signal processing, which is highlighted green in figure 1.2. The main tasks of this design are:

- Digital-Analogue conversion
- Analogue-Digital conversion
- Providing a configurable platform for the controller

The performance of this part is highly dependant on the delay in all respective stages.

The design of this subsystem is documented in a different thesis. See [22] for details.

[2]Because it discusses the project in general, this section occurs in the theses of all three subgroups.

Analogue components
The second part is the design of the analogue system components which are highlighted red in figure 1.2. The aim of this part is designing the motional feedback system for high fidelity applications. Fittingly the amplifier constructed uses active feedback to reduce output distortion. All tasks belonging to this group are:

- Subtracting the controller's error signal from the audio
- Cross-over filtering of the audio
- Amplifying the audio signal with minimal distortion
- Characterising the speaker
- Measuring the displacement of the loudspeaker cone

The design of this subsystem is documented in a different thesis. See [21] for details.

Control and system identification
The final group designs the controller highlighted in blue in figure 1.2 and implements system identification. The control group is tasked with implementing the motional feedback that makes the system closed loop stable and reduces distortion. All tasks of this group are:

- Designing the motional feedback controller
- Implementing system identification that identifies the second order characteristic of the speaker the motional feedback is applied to

Because the system contains system identification it can be applied to all enclosed speakers. This means no redesign is required for installing the motional feedback module on a different second order speaker system.

This thesis will focus on the control and system identification subsystem. For details on the design of the other two subsystems, please refer to [21] and [22].

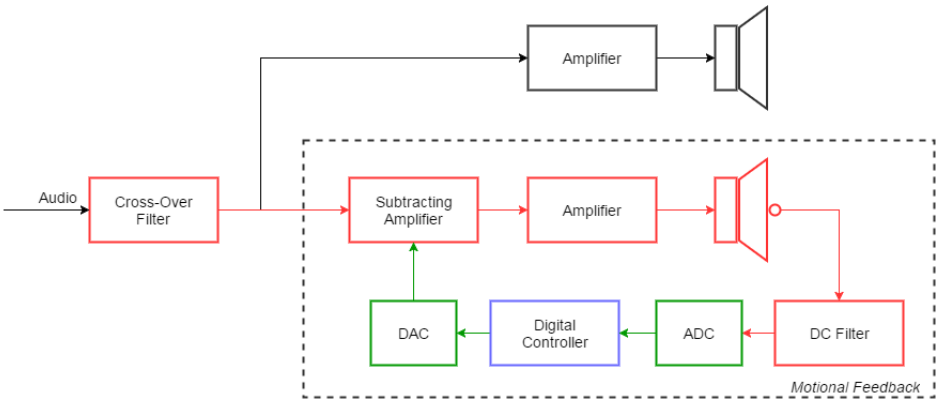


Figure 1.2: The motional feedback system designed by the three groups. The green parts are designed by the digital group. The parts highlighted in red are related to the amplifier/filter group. The blue part is the controller implemented by the control group. The MFB system is shown here in a two-way loudspeaker setup, where the tweeter (on top) does not use motional feedback.

1.3 State-of-the-art analysis

1.3.1 Sensor types

MFB systems have been designed before, most notably by Philips in the year 1968.[3] Their implementation improved the frequency response by flattening it to as low as $30Hz$.

Their feedback system was tested with three types of sensors, namely (1) a displacement, (2) velocity and (3) acceleration sensor. They concluded that the acceleration sensor was most suited for use in a loudspeaker system because it does not require a stationary reference point. Using a displacement or velocity type sensor anyway introduced measurement uncertainties, because the speaker's enclosure (where the reference point was located) vibrated with the sound pressure produced by the conus.

MFB has also been implemented successfully using a velocity type sensor. In [10] the problem of needing a non-stationary reference point was circumvented by not measuring the velocity directly using a motion sensor, but instead by measuring the "cone velocity induced voltage in a secondary speaker coil". Combining this sensor with a speaker with a *single*-wound coil is therefore not possible, while sensors that work with single-wound coils work for both single-wound and dual-wound coils. One of the main advantages of digital MFB is the fact that it can be adjusted to a different speaker on the fly, without the need to redesign the controller for the new speaker (sec. 1.4.4). Using the velocity type sensor of [10] would, because it can be used on dual-wound coils only, halve the flexibility of digital MFB. Furthermore, common issues with velocity type sensors are the temperature dependence of the coil resistance and the non-uniformity of the magnetic field in the coil. These issues can cloud the sensor's measurements and consequently the true picture of the coil velocity.

Taking all of this into consideration, an acceleration type sensor is best suited for this project.

1.3.2 Analogue implementations

Nearly all MFB systems that have been designed implement the controller using electrical circuit components (for example, [3], [10] and [8]). This analogue implementation is very inflexible compared to a digital one. Digital MFB can be adapted to match a new loudspeaker or amplifier, while the analogue system can not be changed once the circuit components of certain values have been soldered in place.

1.3.3 System Identification

Using swarm intelligence to construct non-linear optimisation algorithms is a relatively new development that is the main feature of algorithms like particle swarm optimisation (PSO) [18]. Further optimised adaptions of PSO also exist such as bat swarms [19] and multi-swarm optimisation.
Instead from improving the algorithm itself, some papers investigate the effect of the particle parameters [18]. Depending on the nature of the function to be minimised, this can either improve or decrease the performance of a swarm.

1.4 Problem definition

The problem for this thesis consists of three parts.

Firstly, a controller should be designed that improves the loudspeaker's transfer by using an MFB system.
Secondly, the controller should take advantage of the flexibility offered by a digital implementation as opposed to an analogue one (see section 1.4.4.1).
Lastly, constraints are placed on the controller design by the rest of the system. For example, the system's topology (feedback loop) and the sampling frequency of the ADC and DAC are fixed. For details, see the theses by the other subgroups of this project, [21] and [22].

All three parts of the problem will now be elaborated on.

1.4.1 System characteristics and distortion

A loudspeaker is non-ideal because it distorts sound in multiple ways. When a signal is fed into the system, harmonics start to appear. This is due to the non-linear characteristics of a speaker:

$$y = [\alpha_0] + \alpha_1 u + [\alpha_2 u^2 + \cdots + \alpha_N u^N]$$

The terms between square brackets represent non-linear behaviour of the system. Inserting a sinusoid, for example, produces a sum of sinusoids with N harmonics.
On low frequencies, the mass spring system of the cone starts to resonate and lose its rigidness [3]. Therefore linear distortion is most present in the bass range, where motional feedback will have to aid.
The loudspeaker system, displayed in figure 1.1a by the transfer function G, is combined with the characteristics of the amplifier (D) to create the transfer function of the open-loop system, which will from now on be referred to as DG. G can be approximated as a parallel RLC circuit [3] and thus acts like a first order band pass filter, with one zero and two poles.
For the implementation of the controller, the results of a measurement were used. This version of DG has 2 zeros and 3 poles.

As mentioned in the previous section, motional feedback is shown to be effective in neutralising the effect of both the mass-spring system in the speaker and the imperfections in the frequency behaviour of the amplifier and in reducing the aforementioned distortion.

1.4.2 Sensor and ADC/DAC

By definition motional feedback utilises a feedback loop, i.e. the output of the loudspeaker is measured and used to calculate an error signal that is fed back to correct the input.[3]. Using this configuration, the basic layout of the MFB system will look as in figure 1.1b.

The output of the loudspeaker is sound, i.e. the movement of the loudspeaker's conus. This movement can be measured using (1) a displacement, (2) velocity or (3) acceleration sensor, of which only the third option does not require a stationary reference point. Such a reference point is not easily found in a loudspeaker, so using a displacement or velocity sensor anyway would introduce uncertainties in the measurements due to the non-stationary reference point.[3] Other types of displacement or velocity sensors (that do not directly measure movement) are possible[10] but are not suitable for this project (see section 1.3). The most practical sensor for a loudspeaker is therefore one that measures acceleration, which can also be conveniently mounted on the conus itself.

In figure 1.1b, $y(t)$ represents the acceleration of the loudspeaker cone as measured by a piezo-electric sensor and then digitalised by an Analogue to Digital Converter (ADC) and $e(t)$ represents the error correction signal to be converted in the Digital to Analogue Converter (DAC) and then subtracted from the incoming audio.

1.4.3 Controller

Between the ADC and DAC in figure 1.1b is the feedback transfer unit (K) or "controller" whose goal is to transform $y(t)$ into $v(t)$ with as little delay as possible.[3]

By approximating the open-loop system (DG) as an LTI system with a constant transfer spectrum [3] the closed-loop transfer of the system can theoretically be shaped to any desired shape (H) (see section 1.4.4). The controller

[3]The exact amount of delay (phase shift) allowed will be specified in the Programme of Requirements (see section 2).

is then calculated as:

$$\frac{A}{U} = \frac{DG}{1+DGK} = H \quad \Rightarrow \quad K = \frac{DG-H}{DGH}$$

There are several ways to implement K, given that DG and H, and therefore K(z), are known. Choosing the best implementation of K is part of the problem.

1.4.4 Advantages of digital

The controller should take full advantage of the digital nature of the system.

With analogue MFB the controller is implemented using electrical circuit components that have fixed values and are calculated for a specific loudspeaker and amplifier (DG). This means that if the loudspeaker's (or amplifier's) characteristics vary due to e.g. age, wear and tear or air temperature or humidity changes, the analogue controller won't correct the transfer as good as before and might even make it worse. Tolerances of electrical circuit components might have an adverse effect on the transfer as well.

A big advantage of digital MFB is that it does not suffer from these problems, because it *can* adapt to changes in the characteristics of the loudspeakers. This is done by temporarily switching off the input audio, then determining the actual up-to-date transfer of the system, just before starting the controller. The field of research for this mechanism is *system identification*.

An important consequence of this is that, since the system's transfer is determined anew every time MFB is started anyway, the controller is now independent of the loudspeaker system it is attached to. The same controller and sensor can be used to convert any existing loudspeaker and amplifier into an MFB system – a big advantage for potential business applications!

The controller does not per se have to correct the system's transfer to an ideal, flat frequency transfer. It can be designed to target any (sufficiently smooth) transfer (H), but as before, the analogue MFB system has the disadvantage that it's circuit components have to be calculated for one specific situation, in this case a single transfer H. The digital system's target transfer can be changed at any time. Analogue systems would need a separate equaliser for that.

In short, the main advantage of digital MFB is it's flexibility in handling a speaker's changing transfer characteristics as well as the same controller being compatible with any loudspeaker system. This is achieved using system identification. Using this technique, a digital MFB system is able to create any desired transfer on any speaker system, a big improvement over analogue MFB systems.

System identification is therefore part of this thesis' problem.

1.4.4.1 System identification

The technique of system identification tries to characterise the open-loop system (DG) before MFB is activated, by using some type of test signal as input. Measurements of the system's output are then used to identify its transfer, allowing it to be used to design the controller (sec. 1.4.3).

The identification result can vary from a full, noisy, transfer spectrum to a model consisting of only several parameters of the mass-spring system of the speaker cone and/or the amplifier. This result and the quality of the identification depend on the identification method as well as the test signal used. Choosing a method and test signal are therefore part of the problem.

Functional system identification will mean the open-loop transfer DG can be determined fully automatically. This is a direct result of the system being implemented digitally and a huge advantage over analogue alternatives (see section 1.4.4).

1.5 Synopsis

Now that the problem of designing a digital controller has been defined and already available work on it has been reviewed, it is time to present a programme of requirements, containing both the total system requirements and those specifically for the controller. With these requirements in mind, possible solutions to the problem of the controller subsystem are researched. This includes analysing the stability of the controller, which leads to the requirement of a low-pass filter. The control unit itself can be implemented as either an FIR or an IIR filter, both of which are evaluated as options. Furthermore, notch filters are discussed as a means to counteract the resonance of the closed-loop system.

Next, system identification is discussed by first looking at the various possible test signals and followed by methods of identification. Two methods are looked at in depth, namely linear least squares and particle swarm optimisation.

After this, the solutions are implemented as a prototype and the results of this prototype are discussed. The last chapter details the evaluation of the project, including a discussion of the results, the conclusion and recommendations for future work.

Chapter 2

Programme of Requirements

This section covers the requirements of the system based on the state of the art analysis and limitations of the project itself. First the requirements of the total system are listed. These outline the overall purpose, the time and money restrictions, the input and output, and the criteria of the system transfer. Next the derived criteria for the subsystem designed in this thesis are listed.

1. **Total System**[1]
   - 1.1. **Overall**
     - 1.1.1. The feedback must be implemented digitally.
     - 1.1.2. The system must be able to log its input audio stream and controller parameters for debugging purposes.
     - 1.1.3. The system must be adjustable to the speaker.
     - 1.1.4. The system must work on second order speaker systems.
   - 1.2. **Input/Output**
     - 1.2.1. The maximum peak-to-peak input voltage of the audio signal is 4 V.
     - 1.2.2. The maximum output power is 50 W.
     - 1.2.3. The DC output may not exceed 50 mV.
   - 1.3. **System Transfer**
     - 1.3.1. The gain of the system must be 28 dB ± 1 dB.
     - 1.3.2. The THD must be inside the mask shown in appendix A.
     - 1.3.3. The bandwidth of the total system must be between 10 Hz and 1 kHz.
     - 1.3.4. The SNR is at least 80 dB at the output inside the specified bandwidth of the motional feedback.
     - 1.3.5. The phase margin of the feedback transfer must be at least 45° to ensure stability.
   - 1.4. **Development**
     - 1.4.1. Development time is 9 weeks.
     - 1.4.2. Development costs must be below €300.
2. **Control subsystem**
   - 2.1. **Overall**

[1] Because it discusses the project in general, this section occurs in the theses of all three subgroups (see section 1.2).

2.1.1. The system software must be implemented in the C programming language.

2.1.2. The system must log its computed output as well as the calculated filters' parameters.

2.1.3. The system must be applicable to different hardware setups.

2.2. **Input/Output**

2.2.1. The input and output sample frequency must be four times the bandwidth.

2.2.2. The SNR of the input must be at least 80dB.

2.2.3. The input and output must have a resolution of at least 14 bits.
This follows from specification 2.2.2. and the fact that the average SNR equals the square of the number of quantization levels.[20] Therefore, in decibels, $SNR_{dB} = 10 \cdot log_{10}(2^{2n}) = 10n \cdot log_{10}(4) \approx 6n$ where n is the number of bits. 80dB thus means at least 14 bits are required.

2.3. **System Transfer**

2.3.1. The bandwidth of the controller must be between 10 Hz and 1 kHz.

2.3.2. The transfer must be configurable.

2.4. **Development**

2.4.1. Development time is 9 weeks.

2.4.2. Development of the control system is a software project, so no money can be spent on hardware.

2.5. **Safety**

2.5.1. The system must monitor the sensor output for system instability and switch off the controller in such a situation.

Chapter 3

Solution

3.1 Outline

3.1.1 Solution options

To begin with, the stability of the digital filter K that implements the controller is analysed. It is then elaborated why a low-pass filter is required. The controller itself can be implemented as an FIR or an IIR filter, both of which are analysed as possibilities. Furthermore, notch filters are discussed as a means to counteract resonance of the closed-loop system.
After this, system identification is discussed, starting with the various possible test signals and followed by the method of identification. Two specific methods are looked at in depth, namely linear least squares and a particle swarm algorithm.

3.1.2 State machine

Regardless of the chosen solution, the open-loop system first has to be identified before the controller itself can be turned on. The system therefore functions as a state machine with a state for system identification and one for the active MFB controller.

3.2 Controller

3.2.1 Theory

In the controlling state, the microcontroller will act as a filter to achieve an ideal transfer function. To achieve this, first the basic characteristics of the bass speaker system (including sensor) must be identified: the speaker appears to act as a mass spring system, which means that its transfer function will always have more poles than zeros. A more complete description of the closed loop system can be found in section 3.3.
The system in the controlling state has a setup as displayed in figure 1.1b.
When the transfer function of the bass speaker system (DG) is known through system identification, the controller (K) transfer function can theoretically be determined to set the closed loop transfer to any possible function (H). The transfer function of the total closed loop system is therefore

$$H = \frac{DG}{1 + DG \cdot K} \tag{3.1}$$

which means the controller must satisfy

$$K = \frac{DG - H}{DG \cdot H} \tag{3.2}$$

For the purpose of neutralising bass distortion, H should be set to a constant that is about the average of the open loop transfer (DG) over the sample frequency range (as specified in specification 2.3.1.). In this way, the controller will try to filter out irregularities and most importantly, correct bass frequencies to have an undistorted amplitude. Unfortunately, K can never be applied perfectly. This is because the numerator of DG is a lower order polynomial than the denominator (there are more poles than zeros) and H is a zero-order polynomial (a constant).

$$DG = \frac{\mathcal{P}(n)}{\mathcal{P}(m)} (n < m)$$

$$H = \mathcal{P}(0)$$

Combining this with equation 3.2 and simplifying yields:

$$K = \frac{\frac{\mathcal{P}(n)}{\mathcal{P}(m)} - \mathcal{P}(0)}{\frac{\mathcal{P}(n)}{\mathcal{P}(m)} \cdot \mathcal{P}(0)} = \frac{\mathcal{P}(2m)}{\mathcal{P}(n+m)} \tag{3.3}$$

In order to make K a converging, implementable filter, it can not have more zeros than poles. Equation 3.3 shows that it must then hold that $2m < n+m$. But since for DG $n < m$, it is practically impossible to build a perfect and stable controller filter.
This problem can be fixed by introducing additional poles in the form of a Butterworth LPF.

Summarising, the controller filter must:

- approach the perfect implementation as much as possible
- have as little additional phase shift as possible
- be stable by itself
- not destabilize the total system.

3.2.2 FIR vs IIR and stability

There are three candidates for the implementation of the digital filter: a direct fast Fourier transform (FFT), a finite impulse response (FIR) and an infinite impulse response (IIR).

Direct FFT

The first method that comes to mind is to directly use Fast Fourier Transform. The signal gets recorded in a buffer until the length matches the spectrum of the controller's. Using a Fast Fourier Transform (FFT), the frequency spectrum of the buffer is calculated and then multiplied by the controller spectrum. The resulting complex vector is transformed back to the time domain and written to the outgoing channel.
Advantages of this method are that the spectrum of the controller can be adapted freely per sample, to the most minute details. It can also be obtained directly from measurement data (although noisy), without need for a model identification. Direct FFT is however not a feasible option: the required buffering of the signal causes a large delay in the output, which translates into a linear phase shift. The computation power required to perform two FFTs is also too much for the microcontroller to endure.

FIR filter

A *Finite Impulse Response* (FIR) filter works by means of a convolution with a finite impulse response $h[n]$ in the time domain:

$$K(z) = \sum_{n=0}^{N} k[n]y^{-n} \equiv k[n] * y[n]$$

Here, the time variable n is discrete because the filter will be implemented digitally. The required impulse response $k[n]$ can be obtained by using an inverse Z-transform on the controller's transfer $K[z]$. The digital FIR filter is then built by applying the above convolution.
Implementing the controller as an FIR filter has two big advantages: the filter can always be made stable and a (noisy) impulse response can be extracted directly from measurement data by deconvolution. This impulse response then characterises the system and it need not be fitted to a set of model parameters using elaborate system identification algorithms.
Furthermore, an FIR is not constrained to rational functions and can theoretically take any form, making any desired closed loop transfer a possibility in theory.
However, its main disadvantage makes a conventional FIR filter unusable for a feedback controller. An M-order filter has to perform a convolution on the incoming data to calculate the output, but because the impulse response is symmetrical around $t = 0$, half of it exists before $t = 0$, which means that to determine the output, the filter must look $\frac{M}{2}$ samples into the future. The output therefore has the same problem as a direct FFT implementation: a linear phase delay, or constant time delay of $\frac{M}{2}T_s$ where T_s is the sample time.
An FIR can, however, be adapted to eliminate the delay of the impulse response by simply discarding all data in the left half of the time domain. The consequence is that the effects of the filter leading up to the actual sound will be ignored, but the output of the filter will follow immediately after the input (when ignoring the delay of calculation).
Figure 3.1 shows the approximations of FIRs with and without an anti-causal half, implemented with an order of 256 calculations per sample time. The figure shows that to have a magnitude response that meets the requirements (spec. 2.3.), the filter order should be in the range of 256, which results in too big a phase shift, larger than 180°. A FIR filter is therefore not usable for the controller implementation.

IIR filter

An *Infinite Impulse Response* filter uses recursion of the output to obtain an impulse response of infinite length. This translates to a continuous spectrum in the $\mathcal{Z}$ domain, with an arbitrary number of zeros and poles on arbitrary spots within the $\mathcal{Z}$-plane. In time domain, this looks like:

$$y[n] = \frac{1}{a_0}\left(\sum_{i=0}^{N} b_i x[n-i] + \sum_{j=0}^{M} a_j y[n-j]\right)$$

and its equivalent in the $\mathcal{Z}$ domain looks like:

$$H(z) = \frac{\sum_{i=0}^{N} b_i z^{-i}}{\sum_{j=0}^{M} a_j z^{-j}}$$

This is a rational function, just like most analogue transfer functions. An example would be the Butterworth low-pass filter, which takes many coefficients to implement accurately for an FIR, but only two for an IIR. A recursive digital filter is therefore the most ideal option to model an analogue equivalent. Since the speaker system with the piezo-electric sensor attached is, obviously, an analogue system, the required error correction controller transfer will also represent an analogue system since it was simply derived using equation 3.2.

The IIR has another advantage over FIRs: to obtain an accurate frequency response of any rational transfer function, an IIR requires much less computational power than an FIR. Also, it causes no additional ripple effect. The disadvantage is however that the IIR must be describable by a rational function, which makes spectra like a brick-wall filter very hard to build without switching to an FIR.

Another disadvantage of IIR digital filters is a direct cause of their recursion: wheras it creates the ability to place poles and zeros anywhere in the $\mathcal{Z}$-plane, it also makes poles outside the unit circle a possibility. Other than the last two solutions, an IIR can therefore be unstable by itself. This has to be taken in account when designing it in the system identification phase.

Lastly, the phase response of an IIR filter is generally not linear, which will cause phase distortion. Fortunately this is beneficial for the feedback controller implementation, since the controlled phase response of K can be used to neutralise the phase distortion of DG.
The efficiency of an IIR is visualised in figure 3.1. With an order of only seven calculations per sample time, the spectrum approaches its analog counterpart almost perfectly until the highest frequencies.

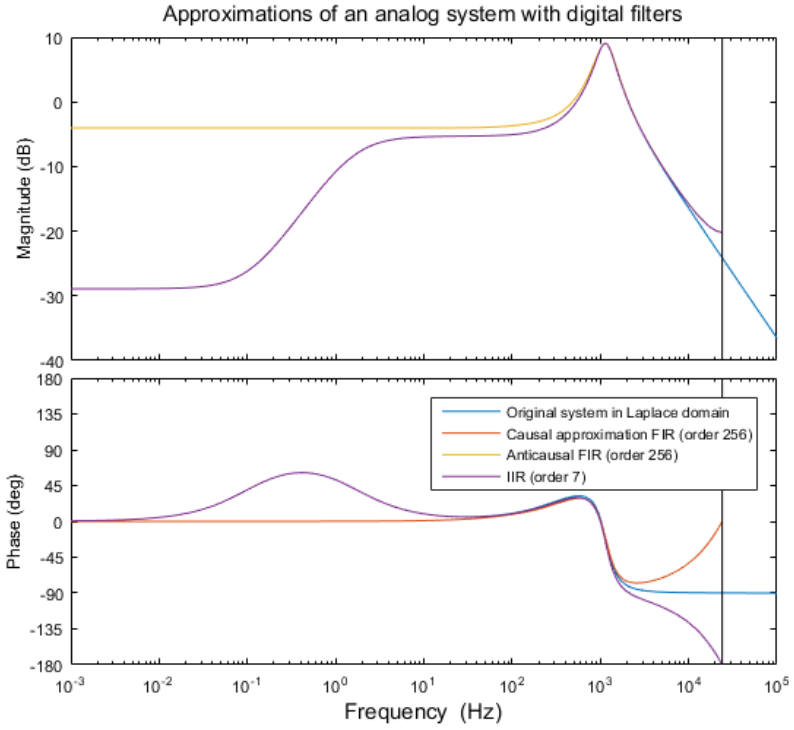


Figure 3.1: Bode plots of digital filter implementations of an FIR, a causal approximation FIR and an IIR. The approximated spectrum is of a transfer function in the Laplace domain and is plotted in blue. The phase response of the anticausal FIR filter (yellow) is so large that it was not displayed in this plot.
The bandwidth of interest is between $10Hz$ and $1kHz$ (spec. 2.3.1.), but the full spectrum is shown to be able to spot instability issues due to the transfer outside of the controller's bandwidth.

3.2.3 Low-pass filters

It was already shown that any implementation of the base controller can not be made stable. To fix this, an LPF in the $\mathcal{Z}$ domain can be multiplied with the base controller to introduce enough poles to stabilise it. The order of the LPF can be adapted to increase the order of the denominator to that of the numerator. There is however a disadvantage: the LPF decreases the effect of the controller and causes a slight additional phase shift at higher frequencies. This can still lead to an unstable total system, therefore the -3dB frequency of the Butterworth filter must be optimised to achieve the most straight total system transfer. For this matter a -3dB frequency optimisation algorithm is implemented.

3.2.4 Notch filters

Reason for notch filter

The sensor is made of a piezo-electric material, which will resonate at a certain frequency. For the specific sensor used for testing this resonance frequency is within the bandwidth where the controller should work. More specifically, the resonance frequency of the tested sensor was measured to be roughly $1kHz$, which is the upper bandwidth requirement of the controller (see requirement 2.3.1. in chapter 2), so the resonance will affect the transfer within the target bandwidth. For this reason, and to prevent the resonance from destabilizing the closed-loop system, the peak should be flattened.

Approach and requirements

To flatten the peak, it is first determined (1) at what frequency it occurs (ω_0) and (2) what it's magnitude is at this frequency ($|H(j\omega_0)|$). The goal then becomes to design a notch filter that counteracts this peak. Requirements of the notch filter are:

1. Apart from at the notch itself, the frequency response is flat and has a gain of $0dB$ to not change the system's frequency response unnecessarily. This means the poles and zeros lie in a circle around the origin in the s-plane. For the notch to be stable, this is the semi-circle in the left half-plane.
2. The notch is causal, so all poles and zeros come in conjugate pairs.
3. The notch should be the simplest possible. Because at least one pole and one zero are needed to make a notch and poles and zeros come in conjugate pairs, two poles and two zeros are used.
4. The notch is centered at ω_0, which means the semi-circle has radius ω_0.
5. The magnitude of the notch is the desired magnitude H_0 divided by the resonance peak's magnitude $|H(j\omega_0)|$. In decibels it is the difference between the two.

These requirements result in the pole-zero map of figure 3.2. The zeros are denoted by a_0 and a_1 and the poles by p_0 and p_1.

Location in system

The notch filter designed by the approach outlined above flattens the resonance peak in the transfer of the closed-loop system. This means it should be placed directly after the closed-loop system. However, as can be seen in figure 1.1b, there is no place for a filter there in the hardware setup used for this project. All control systems and filters should be implemented digitally (spec. 2.1.1.) on the microcontroller board. This is done by altering the desired transfer H_0 as if the notch filter ($NOTCH$) was placed at the output of the closed-loop system. The new

desired transfer is given by equation 3.4.

$$H_{0,new} = H_{0,old} \cdot NOTCH \tag{3.4}$$

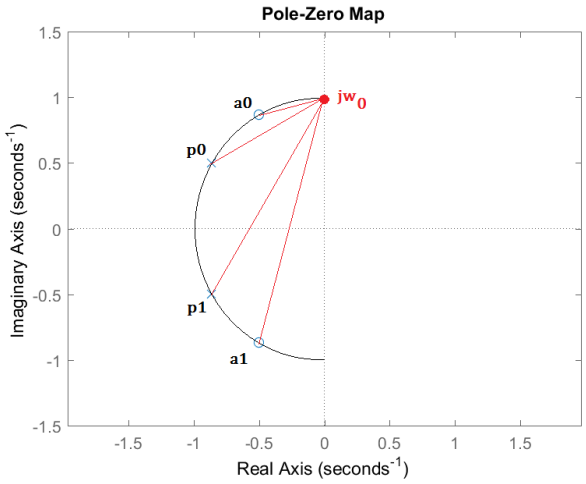


Figure 3.2: Pole-Zero map of notch filter. The zeros are denoted by a_0 and a_1 and the poles by p_0 and p_1.

3.3 System identification

3.3.1 Test signals

To identify the loudspeaker system, we need a test signal that has most of its power centred around the frequencies where the system operates, so in this case the lower frequencies. There are a few options: [13]

Delta pulse A delta pulse is one single unit impulse of a very short duration. Because of it's infinitely steep edges, it contains all frequencies and might sound like an ideal test signal. Containing all frequencies would however also mean that the signal has infinite power, which is impossible in the physical world. The real-world delta pulse will be slightly smoothed and spread out by for example the DAC used. This means that an ideal delta pulse can not be offered to the system as an input.

Gaussian white noise Gaussian white noise has a constant power spectrum over all frequencies, but because it is random, and therefore uncorrelated, there is no way of retrieving the test signal from the measured transfer.

Pseudo-random sequence If the resulting values of Gaussian white noise are determined and stored, however, the signal can be retrieved from the measurement. The sequence is then pseudo-random. In this way the flat spectrum of Gaussian noise can be used as a test signal.

Sine stack A sine stack has a discrete set of impulses in its spectrum, instead of it being constant. The advantage over Gaussian white noise is that the test signal can be defined.

3.3.2 Identification methods

Methods for system identification can be either parametric or non-parametric. It may seem advantageous to opt for a non-parametric method, since it is then not needed to construct a parametric model of the system beforehand. When the order or type of a system is unknown, the measurements may be approximated by any fitted curve independent of the number of parameters [16]. However, when information about the model is known, for example using physical insights as in our case, it is beneficial to use this information to define the model parametrically before applying further system identification.[5] Therefore, the parametric approach is expected to yield the best results with limited computational resources, as in our case. Non-parametric methods are thus not pragmatic for our application of speaker system and not elaborated further below.
When a set of noisy data is obtained from the test signal, a fitting curve needs to be identified. The transfer function of any system can be described by:

$$F(s) = C \cdot \frac{(s-a_0)(s-a_1)\cdots(s-a_n)}{(s-p_0)(s-p_1)\cdots(s-p_m)} \tag{3.5}$$

with n the number of zeros and m the number of poles.
The mass-spring system can be modelled by the mechanical equivalent of an RLC oscillator, or a first-order band pass filter [3]. Therefore the system is estimated to have two poles, and one zero. Since it is known how many poles and zeros the audio system has, the amount of parameters is also fixed. This allows for parametric system identification. There are a number of ways to perform this, but we will focus on the method of least squares and particle swarms optimization.

3.3.2.1 Least squares

The method of least squares tries to minimise the squared residue of N data points with the estimating function:

$$S = \sum_{i=1}^{N} (y_i - f(x_i, \beta_1, \beta_2, \ldots, \beta_j))^2$$

$$\frac{\partial S}{\partial \beta_j} = 2\sum_{i=1}^{N} (y_i - f(x_i, \boldsymbol{\beta})) \frac{\partial f(x_i, \boldsymbol{\beta})}{\partial \beta_j} = 0$$

In order to define the parameters in $\boldsymbol{\beta}$ analytically, it must be possible to explicitly define them. The fitting function f must therefore be linear to achieve this:

$$f(x_i, \boldsymbol{\beta}) = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_M x_i^M$$

because differentiating to a parameter β_j must result in a function of only x_i: then the resulting set of M equations can be solved with a matrix equation.
Unfortunately, the function the data needs to fit to (equation 3.5) is far from linear and still has a lot of parameters stuck in the formula after differentiating to any parameter. Therefore, exactly and explicitly deriving the least squared error fit to the data measured by the microcontroller is impossible. However, there are ways to approach this minimum error value numerically, for example by using a particle swarm optimisation.

3.3.2.2 Particle swarm optimisation

Particle swarm optimisation, or PSO for short, is a relatively new technology that utilises swarm behaviour to approach a minimum in a search space of an arbitrary dimension. It uses a swarm of virtual “particles” that communicate with each other but also fly around the search space in a random fashion. Based on the philosophy of swarm intelligence, derived from natural phenomena like fish schools or bird flocks, the particles will converge

to the minimum of a given function given enough time and numbers, just like birds will converge to a tree or fish converge to bait.
Digitally this social behaviour can be simulated by the PSO algorithm [17]:

1. Initialise
   (a) the particles receive random positions ($\mathbf{x_i}$) divided uniformly over a search space with bounds $\mathbf{b_{lo}}$ and $\mathbf{b_{up}}$.
   (b) the particles receive a random velocity ($\mathbf{v_i}$) within the same search space.
   (c) the previous best position of every particle ($\mathbf{p_i}$) is set to the current (and first) position.
   (d) the best position of the entire swarm ($\mathbf{g}$) is set to the smallest value of ($\mathbf{p_i}$)
2. Iterate
   (a) For every particle and dimension, calculate a new velocity depending on the current particle variables and a few factors [18]:
      - the current velocity, dependent on the inertia weight c_v
      - the previous best, dependent on the cognitive scaling parameter ϕ_p and a uniform random variable $r_p \in [0,1]$
      - the swarm best, dependent on the social scaling parameter ϕ_g and a uniform random variable $r_g \in [0,1]$
      - the current particle position.
   (b) Move the particle by adding the velocity to the position.
   (c) Constrain the particles in the search space, if they have moved out of it.
   (d) Is the new position a better fit than the last position? $\rightarrow$ update the previous best position.
   (e) Is the new previous best a better fit than the current swarm best? $\rightarrow$ update the current swarm best.
3. Terminate after I cycles or when the swarm best passes a maximum margin.
4. $\mathbf{g}$ now holds the closest fit to the data.

While relatively intensive in calculation, because of the multiple function calls per particle per iteration to determine the best position, this numeric approximation technique has proven to work quite efficient in estimating the minimum square error. Since time is not a priority for the system identification application it was found to be a fitting alternative to the linear method of least squares.

Chapter 4

Implementation

4.1 Prototype

4.1.1 Full system and state machine

The MFB system should perform system identification before turning on the controller that performs the actual motional feedback. To this end, a state machine with two states is implemented, the states being:

0. System identification
1. MFB control

The transitions between these states are detailed in the state diagram in figure 4.1. Implementing these states in the actual system requires some additional hardware. Directly after the audio input there is a relay that switches to ground when the system's state is 0, so that the user can not accidentally disturb the system identification phase by already sending a signal through the system. The microcontroller board[1] is displayed as containing two separate parts: the system identification part that emits the test signal and measures the outcome and configures the digital filter K and the part that simply implements said digital filter as the controller. The values of the signals determined by the state machine are displayed in table 4.1.

Table 4.1: States of the Finite State Machine and the corresponding values of the signals. $k[n]$ and $k(t)$ represent the impulse response of the controller K.

State	$u(t)$	$e(t)$	$w(t)$
0 = measure	0	$-m(t)$	$m(t)$
1 = active	$u_0(t)$	$k[n] * a[n]$	$u_0(t) - k(t) * a(t)$

4.1.2 Controller

The controller will eventually be implemented on a microcontroller (spec. 2.1.1.). Before that, however, MATLAB and Simulink were first used to design and test the controller's efficiency and stability. MATLAB was used to determine the characteristics of the closed loop system and the controller mathematically, while Simulink served as a confirmation method by simulating the filters numerically. The Simulink model was composed of two discrete filters in a closed loop setup as displayed in figure 4.3.

[1] The digital controller is implemented on a microcontroller board, but the details of this design choice are outside the scope of this thesis. See [22] for further details.

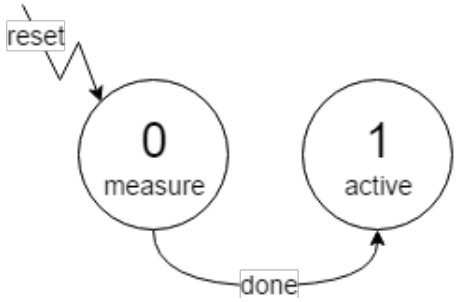


Figure 4.1: State diagram of the controller. During "measure," the system first sends a test signal through the open loop system, then buffers it, and then performs a model identification. During "active," the digital filter provides a feedback controller.

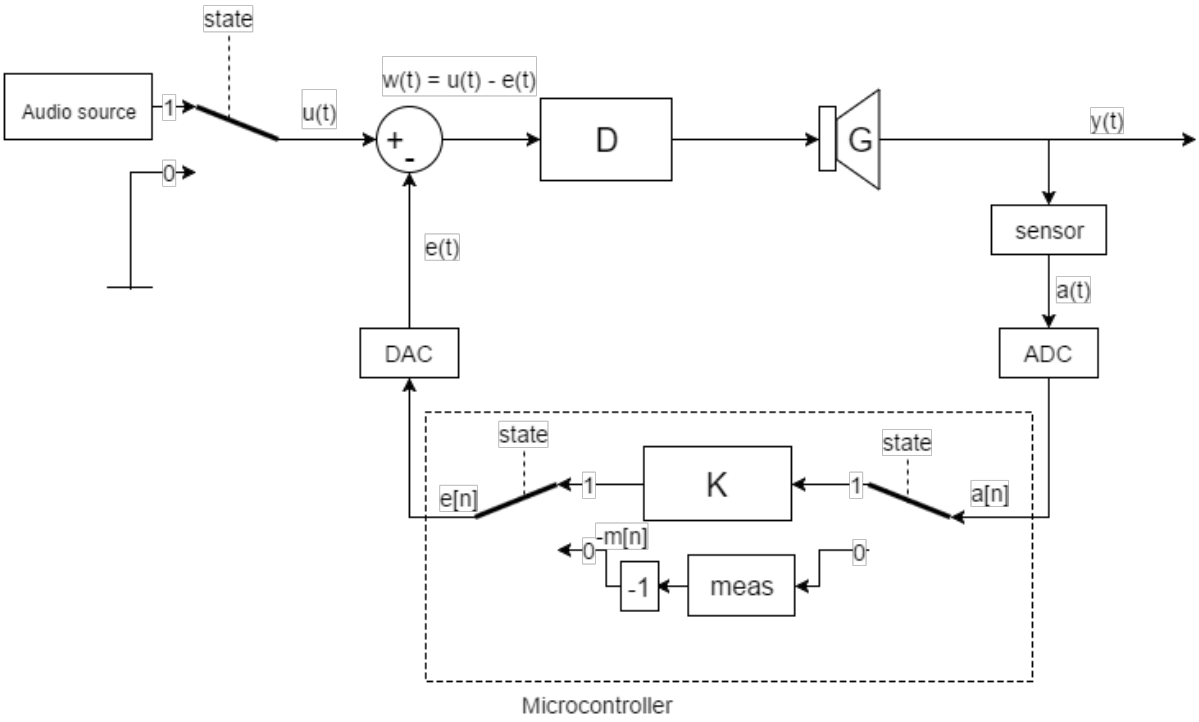


Figure 4.2: Layout of the full system. Shown are the loudspeaker G, amplifier D, controller K, audio input signal $u(t)$, speaker output pressure $y(t)$, conus acceleration $a(t)$ and $a[n]$ and the error signal calculated by the controller $e(t)$ and $e[n]$. The test signal from the system measurement block is $m[n]$.

The open loop transfer DG was modeled as an IIR filter with a transfer function that was measured from a testing loudspeaker. The controller K was either an FIR or an IIR, with a Butterworth LPF of varying order. As a testing signal a digital impulse was used, and the result was transformed to the frequency domain using Fast Fourier Transform (FFT).
In MATLAB, the measurements of DG were loaded and converted to the $\mathcal{Z}$ domain. The desired transfer function H_0 was set to a constant value and the base controller transfer K_0 was determined using equation 3.2.

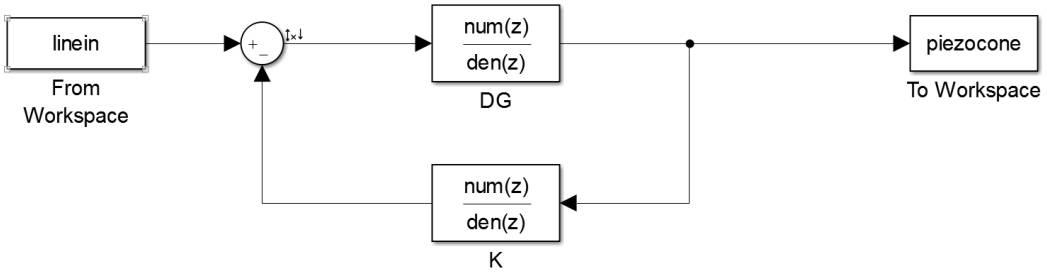


Figure 4.3: The Simulink model setup for closed loop analysis.

Using an FIR filter

FIR filters are inherently stable. While under the effects of ripple and not really fit for analog spectra, the total closed loop has the best chance of stability with a causal approximation FIR filter. An LPF is still required to ensure the inverse Laplace transform to obtain the impulse response is convergent (see also equation 3.3).

Using an IIR filter

IIR filters can not always be made stable. They do, however, very accurately model systems in the Laplace domain. By equation 3.3 a perfect implementation is inherently unstable, so there must once more be an additional pole to counteract the surplus of zeros. After the system identification stage the system runs an LPF optimisation algorithm.

LPF optimisation algorithm

Before the controller is implemented, a fitting LPF needs to be found. The LPF attenuates the frequencies that will cause positive feedback and destabilise the system. The filter causes the poles that fall outside of the unit circle to shift inwards. But it also reduces the effect of the controller in higher frequencies of the closed loop transfer, meaning that the eventual transfer will converge from the desired constant to the original open loop transfer. That is why the -3dB frequency of the LPF should be chosen as high as possible without compromising the stability of the closed loop system, in order to at least retain its positive effects in the lowest bass regions of the loudspeaker. The LPF optimisation algorithm takes care of that.

Looping through a set of suggested -3dB frequencies, the script calculates the transfer function of the closed loop (H) in the $\mathcal{Z}$ domain and retrieves the locations of the zeros and the poles numerically. It then checks the absolute values of the poles. If a value is larger than one, the closed loop is unstable and the LPF needs to be shifted further left in the spectrum. The algorithm loops until all poles lie within the unit circle.

Now that the characteristics of the LPF filter are determined, it is applied to the base controller by multiplication in the $\mathcal{Z}$ domain.

Notch filter

Since the largest absolute value of poles of the closed loop after LPF optimisation is close to 1, the magnitude response of the system will show a resonance peak. To counteract this peak as well as the original sensor resonance peak, a notch filter is added. Following the approach outlined in section 3.2.4, the notch filter can be constructed by placing the poles and zeros according to the requirements mentioned in that section.

In other words, the poles and zeros should now be placed in such a way that the filter has the desired magnitude at the specified frequency. The relationship between the magnitude at a frequency and the pole and zero locations

is given in equation 4.1. The magnitude equals the product of (inverse) distances in the complex s-plane between the frequency $j\omega$ and the poles and zeros. For easier reading, let the magnitude at frequency ω be denoted by $\text{mag}(\omega) = |H(j\omega)|$.

$$\text{mag}(\omega) = \frac{|j\omega - a_0| \cdot |j\omega - a_1|}{|j\omega - p_0| \cdot |j\omega - p_1|} \tag{4.1}$$

Because of the symmetry about the real axis and the fixed radius of the semi-circle, there are only two unknowns: the angles of a_0 and p_0. Using this knowledge equation 4.1 becomes:

$$\text{mag}(\omega) = \sqrt{\frac{(\omega^2 + \omega_0^2)^2 - 4\omega^2 \text{Im}^2(a_0)}{(\omega^2 + \omega_0^2)^2 - 4\omega^2 \text{Im}^2(p_0)}} \tag{4.2}$$

Calculation resulted in the imaginary parts of the pole $\text{Im}(p_0)$ and zero $\text{Im}(a_0)$ as unknowns instead of the angles, which defines the pole and zero's locations equally well.

To solve for two unknown variables, equation 4.2 has to be evaluated twice at different frequencies. One obvious choice of frequency is ω_0, the location of the resonance peak of which the frequency and magnitude is known. At $\omega = \omega_0$ the magnitude simplifies to:

$$\text{mag}(\omega_0) = \sqrt{\frac{\omega_0^2 - \text{Im}^2(a_0)}{\omega_0^2 - \text{Im}^2(p_0)}} \tag{4.3}$$

Solving this equation for $\text{Im}(a_0)$ yields:

$$\text{Im}(a_0) = \sqrt{\omega_0^2(1 - \text{mag}^2(\omega_0)) + \text{mag}^2(\omega_0)\text{Im}^2(p_0)} \tag{4.4}$$

A second frequency of choice (ω_1) at which the magnitude is known has to be chosen. This second constraint on the filter's frequency response is not only necessary to solve for the two unknowns, but also gives the possibility to specify the shape of the notch in more detail. For example, the width of the notch at $3dB$ above the peak could be specified.

Substituting 4.4 into equation 4.2 evaluated at ω_1 yields an equation for the second unknown:

$$\text{Im}(p_0) = \sqrt{\frac{\text{mag}^2(\omega) - 1}{\text{mag}^2(\omega) - \text{mag}^2(\omega_0)} \cdot \left(\frac{\omega^2 + \omega_0^2}{2\omega}\right)^2 - \frac{\text{mag}^2(\omega_0) - 1}{\text{mag}^2(\omega) - \text{mag}^2(\omega_0)} \cdot \omega_0^2} \tag{4.5}$$

Using these two explicit formulas (eq. 4.4, 4.5) for the unknowns, the locations of all poles and zeros are fixed, and the design of the notch filter is completed. The filter is then applied to the system by using equation 3.4 to alter the desired system transfer.

4.1.3 System Identification

The system identification stage happens before the LPF optimisation stage. In this stage, the controller first sends a test signal via the error correction port, multiplied in the digital time domain by a factor -1 so that the test signal simulates an input from the line in. The controller has an FFT obtained spectrum of the test signal readily available. When the test signal is emitted, the controller immediately starts measuring the acceleration of the cone via the piezoelectric sensor, and thus obtains a response to the test signal. This response is then deconvoluted by division in the frequency domain with the test signal and a transfer spectrum of DG is derived.
Since this is still a set of raw data, it is not yet of any use to the controller. Depending on the estimated order of the polynomials in the transfer function fraction, a system identification is performed, that returns a more simplified version of the spectrum that fits the raw data best.

Model identification

The implementation of the PSO algorithm is very similar to the theoretic implementation. If the best position corresponds with a function value that is larger than this maximum, the swarm might have landed into a local minimum or moved too slowly towards the solution to reach the global minimum before the iterations were done. In that case the algorithm discards the entire swarm and resets the script. If the particles converged to a local minimum which proved to be insufficient, they are scattered once again.
According to the Standard Particle Swarm Optimisation version from 2007 (SPSO2007) the ideal size for a particle swarm is [17]

$$S = 10 + 2\sqrt{\texttt{dim}}$$

To define the maximum allowed minimum, the standard deviation σ of the measured data spectrum is used. The search space $E \in [\mathbf{b_{lo}}, \mathbf{b_{up}}]$ was defined as 4-dimensional ($\texttt{dim} = 4$), with transfer function numerator and denominator orders of respectively 1 and 2. The four corresponding parameters that define the transfer function are:

- K: the constant before the transfer function. $E \in [0, 1000]$
- a_0: the value of the zero. $E \in [0, 1]$
- p_r: the absolute value of both poles. $E \in [0, 1]$
- p_ϕ: the absolute phase of both poles. $E \in [0, \frac{\pi}{2}]$

These dimensions can be realized given a few assumptions:

- The open loop system is stable and minimum phase: there are no zeros or poles outside the unit circle.
- The open loop system is real: the impulse response has no imaginary parts, since the sensor cannot measure imaginary values, so its $\mathcal{Z}$ transform is symmetrical around the real axis.
- The open loop system is underdamped: the poles are both complex and complex conjugates.

Given these parameters, the data fit will look like this:

$$z = e^{j\omega_i/F_s}$$

$$f(z, C, a_0, p_r, p_\phi) = C \cdot \frac{(z - a_0)}{(z - p_r e^{jp_\phi})(z - p_r e^{-jp_\phi})} \tag{4.6}$$

The three parameters c_v, ϕ_p and ϕ_g were found to be most efficient when set to $c_v = 0.9$; $\phi_p = 0.6$; $\phi_g = 0.1$. The large value of c_v results in a global search rather than a local search, the large value of ϕ_p ensures relatively faster convergence and a small ϕ_g to prevent clustering of the particles [18].

4.2 Results

4.2.1 Controller

Figure 4.4 shows the transfer functions resulting from a simulation in Simulink and mathematical deduction in MATLAB. The closed loop transfer (H), the ideal closed loop transfer (H_0) and the open loop transfer (DG) are all defined as a $\mathcal{Z}$ domain transfer function (tf type in MATLAB), and the Simulink transfer was obtained by applying FFT to the impulse response of the closed loop system. The open loop transfer (DG) was provided by the subgroup that performed measurements of the speaker[21] (see also section 5.4). A model was fitted to the data using MATLAB's function tfest for the purpose of testing the controller. This was done to be able to test the controller using this DG seperate from the system identification algorithm, so that the algorithm's performance

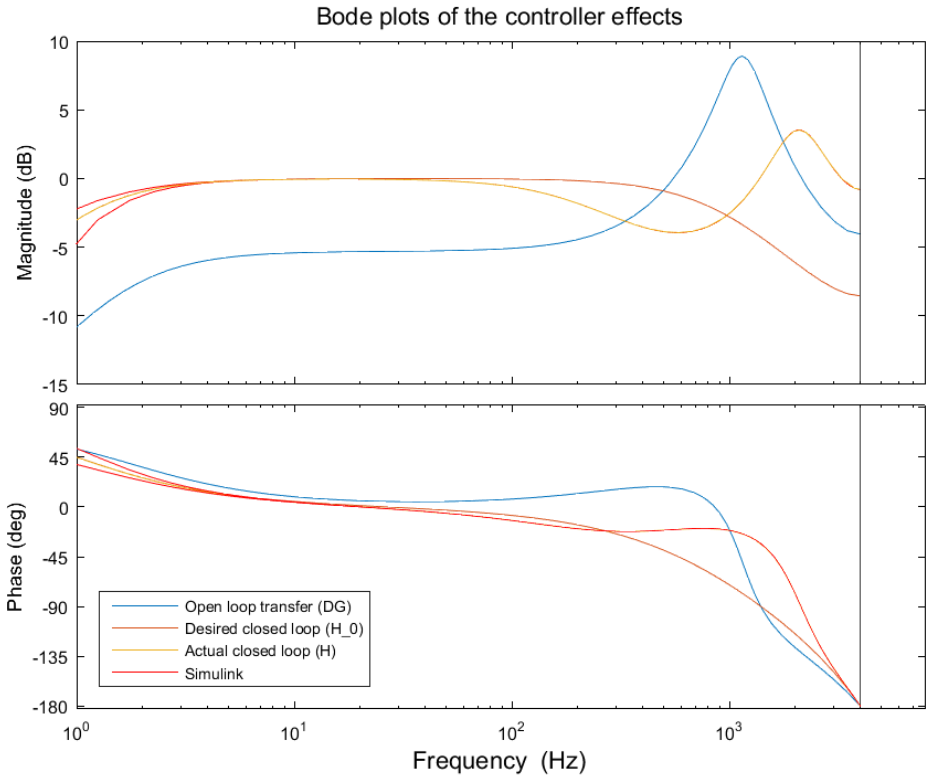


Figure 4.4: Bode plots of the actual and ideal closed loop system H. The graph also shows the original open loop system DG and the result of a Simulink simulation. The peak in *DG* just above $1kHz$ can be explained as the piezo-electric sensor's resonance. No notch filter was applied to the controller yet (see section 5.4).
Note the influence of the added LPF. Without it, the actual closed-loop would be exactly the desired closed loop, but the controller would be divergent. Including the LPF in the controller, the controller is convergent but it's effects are reduced at higher frequencies, as can be seen here: the actual closed loop resembles the desired closed loop less, and resembles the open loop more.

does not influence the controller's performance. The peak just above $1kHz$ can be explained as the piezo-electric sensor's resonance.
Figure 4.5 shows pole-zero maps of the open loop system, the controller (with LPF), the LPF and the resulting closed loop system.

4.2.2 System Identification

When the PSO algorithm is initiated, the search space looks like displayed in figure 4.6
The test setup has the original system with additional white Gaussian noise over the real and imaginary parts of the spectrum. The particles must then try to approach a minimum square error. The result is displayed in figure 4.7.

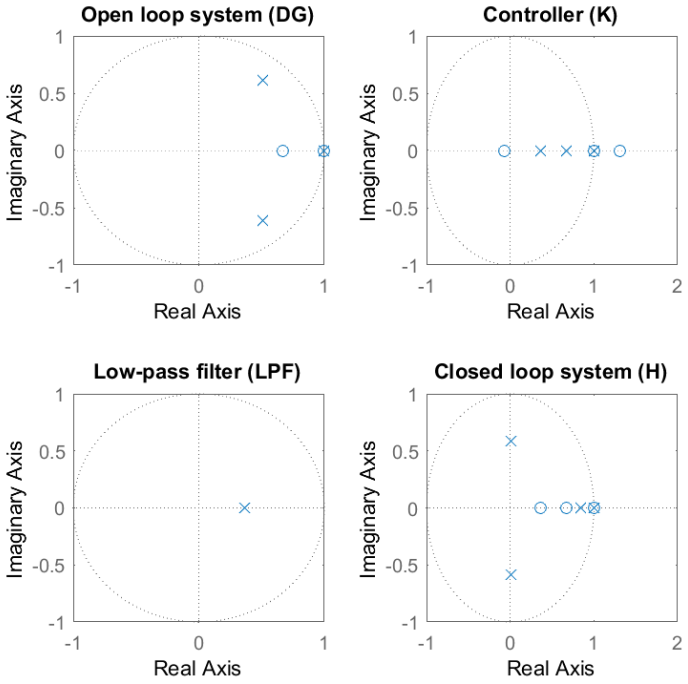


Figure 4.5: Pole zero maps of DG, K, LPF and H.

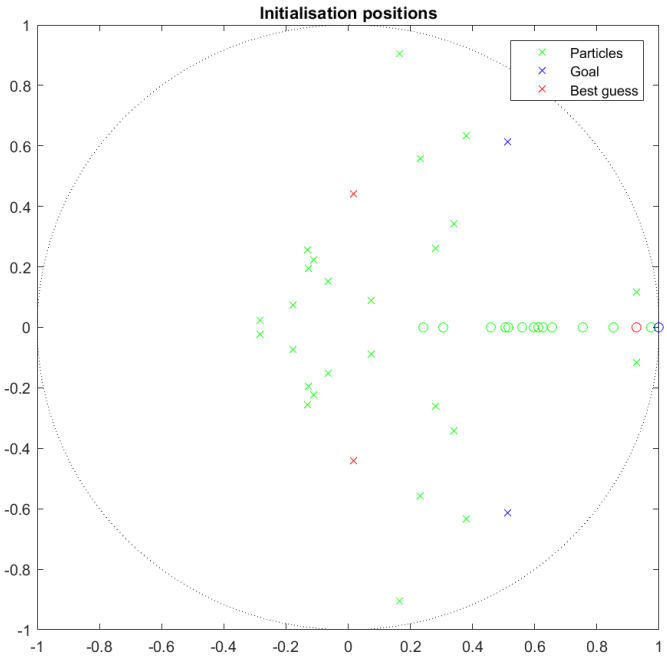


Figure 4.6: What the search space looks like after a PSO initialisation. The green poles and zeros are the uniformly distributed solutions provided as "particles", the red poles and zero are currently the best fitting solution and the blue poles and zero are the original system.

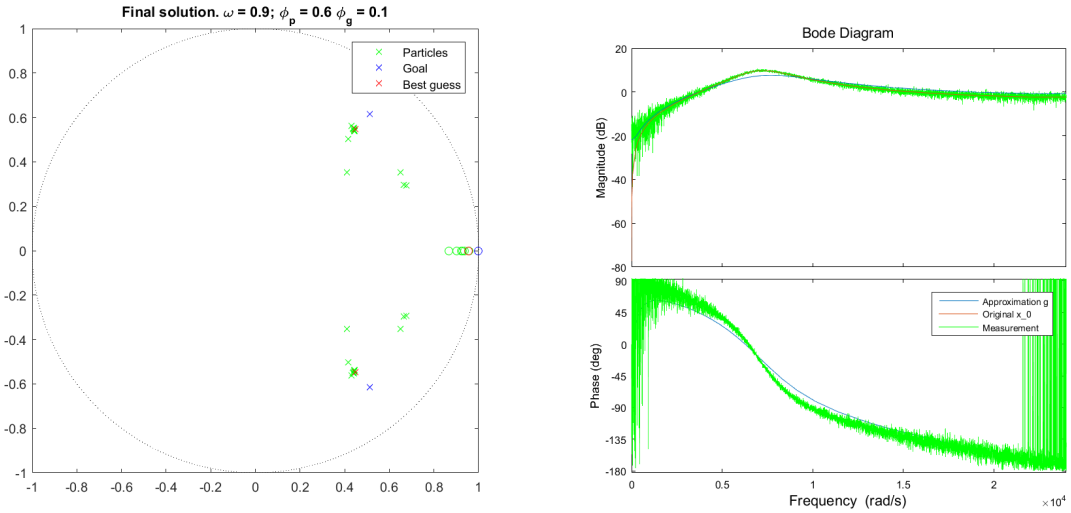


Figure 4.7: A pole-zero plot of the solutions provided by the 14 particles. To the right of the graph there is a bode plot of the original system, the original system with added Gaussian noise and the estimation of the PSO algorithm.

Chapter 5

Evaluation

5.1 Discussion of results

5.1.1 Controller

The actual closed loop transfer displayed in figure 4.4 is almost the same as the desired loop transfer within a margin of ± 5 dB. Therefore, the magnitude response approaches the desired closed loop transfer sufficiently. The Simulink model (in red) confirms the calculated closed system transfer. The resonance peak of the original open loop system is reduced and shifted out of the required bandwidth, so no notch filter seems necessary.
This is probably due to the fact that the open loop transfer was assumed an ideal, third-order LTI system described by a Laplace transfer function. In reality, the spectrum of the speaker system will show more irregularities such as resonant frequencies and is not perfectly time invariant. Also, quite a lot of the distortion will be non-linear. However, the system identification phase currently has to approximate the open loop system with a linear transfer function, so now only linear distortion can be neutralised.

5.1.2 System identification

While quite efficient in most cases, the PSO algorithm has a randomly decided tendency to converge on local minima, which is still a problem, because the random nature of the particle swarm can not yet be controlled. Said problem was solved temporarily by setting a requirement for the obtained model parameters, but unfortunately this requirement is hard to determine effectively.
The computational power is also an issue. A better chance of finding an optimal fit quickly requires a lot of calculations since the swarm size has to become large.

5.1.3 Integration with overall system

Some specifications concerning the integration with the overall system (see chapter 2) were not met or tested for compliance with the Programme of Requirements. For example, the system should be implemented in the C programming language (spec. 2.1.1.). To test this, however, the project's subgroup that works on the microcontroller first has to provide their platform to us, the controller subgroup. Since the microcontroller group had too much delay in their system and not enough time to fix this, this specification was not tested. Therefore, data logging (spec. 2.1.2.) was not implemented either.

5.2 Conclusion

5.2.1 Total system

The concept of motional feedback has been around for almost half a century, but with the current technological advancements in digital applications it is now possible to improve the analogue implementation with a single digital system. Using a relatively cheap microcontroller compared to a more costly custom-built analogue filter system, linear distortion in the lowest regions of a bass speaker can be neutralised. The system is applicable to any bass speaker with a piezoelectric sensor attached thanks to the system identification algorithms. Using digital filters also removes constraints of physical components and allows the speaker to assume virtually any transfer function within the $\mathcal{Z}$ domain.

5.2.2 Controller

The controller is the primary functional part of the feedback system. Several ways to approximate the required controller spectrum to adapt the open loop transfer function have been evaluated. Direct FFT was quickly proven to be unfeasible for application in a feedback loop. FIR filters appeared to have the same problem, but when the anti-causal part was ignored, they still performed relatively sufficiently. But the best implementation yet was the IIR, because of its ability to model an analogue system very accurately at lower frequencies, with a filter of a much lower order.

By mathematically deriving the numerator and denominator of the IIR filter, with an additional LPF to account for the surplus of zeros, the model could be simulated using Simulink. The LPF was optimised to stabilise the closed loop with the help of an optimisation algorithm.

A generic notch filter was designed that does not require an iterative design procedure but can be constructed by using only explicit formulas. The frequency, magnitude and shape of the notch can all be specified exactly, which allows the use of this filter design to compensate for (resonance) peaks in the system's transfer.

Eventually the filter had a significant effect on the closed loop transfer. In theory, the desired band-pass filter with more power in the lowest frequencies could be approached within just a few decibel. In practice, the speaker system will be harder to control, but resonance peaks and notches in the transfer can be fixed by the bandpass, bandstop or notch filters.

5.2.3 System Identification

The ability to fully automatically perform a system measurement and model estimation is the feature of the digital implementation of MFB that is most appealing. Starting with channel estimation, the microcontroller will send a testing signal through the speaker system to retrieve a complex and noisy spectrum.

The rational function that is hidden in the noisy complex spectrum must be derived using the method of the smallest squared error, or least squares optimisation. Unfortunately, the linear approach is incompatible with rational functions like a transfer function. Instead, a numerical approach is used.

Particle swarm optimisation is a relatively novel technology that uses swarm intelligence to approach the minimum of an arbitrary function with an arbitrary number of parameters. Whereas it requires quite some computational power, it approximates the transfer function of the open loop accurately in most cases. The random nature of the algorithm however makes it hard to control, so a verification condition must apply.

5.3 Recommendations

To calculate the closed loop transfer in the $\mathcal{Z}$ domain, the MATLAB tf (transfer function) format was used in combination with the c2d command, to convert continuous (Laplace domain) transfer functions into $\mathcal{Z}$ functions.

MATLAB can use several techniques to sample the impulse response to become discrete, such as zero order hold. Unfortunately none of these techniques are perfect and the poles and zeros end up placed slightly off target. This has the consequence that when analytically a pole should divide away a zero, they end up shifted over each other instead of disappearing. Thus poles and zeros can land outside the unit circle, destabilising a filter when they should not have been there in the first place.

The solution for this is to analytically derive the closed loop transfer function, dividing away as many terms as possible and only then translating to the $\mathcal{Z}$ domain, instead of letting MATLAB do the work. Alternatively, the c2d function could still be used, but with careful consideration which of the conversion techniques is best suited in every new situation the command is used, along with the downsides of it.

5.4 Future work

The most prominent issue with the linear approximation approach of system estimation is that non-linear distortion does not get characterised. Due to negative feedback the harmonics are attenuated to a small extent, but improvement of model estimation or using a different, more unconventional approach to the controller might tackle the problem of harmonics more efficiently.

Since there is not a readily available, ideal method for non-linear least squares optimisation like there is for its linear counterpart, there are numerous possibilities to estimate a rational parametric model. Even within the realms of PSO much research is put into making the algorithm as efficient as possible. Another quicker or simpler numeric estimation might be available.

More specifically, future work might include using a model for the open-loop system that more closely resembles the actual measured open-loop system. On that account, improvements to resulting identified system are desirable, with a focus on the lower frequencies. Furthermore, a different, preferably sharper, low-pass filter might be considered instead of the Butterworth filter currently used. A sharper filter might mean it's cut-off frequency can be higher, resulting in a controller that deviates less from the ideal controller. Lastly, the method of moving the notch filter inside of the feedback loop (eq. 3.4) does not provide the desired flat, notch-free response the notch was calculated for, because equation 3.1 is not linear in K. Instead of equation 3.4, which either compensates the resonance peak too much or too little, it is desirable to design a better method to move the notch filter into the feedback loop.

The implementation of the system would benefit from some additional work as well. The points mentioned in section 5.1.3 can be improved upon. For example, the controller should be implemented in the C programming language and it should support data logging. Once the system can be tested using hardware, tests could be performed to measure the SNR and resolution of the controller's input and output (spec. 2.2.3. and 2.2.2.).

Appendices

Appendix A

Distortion Mask

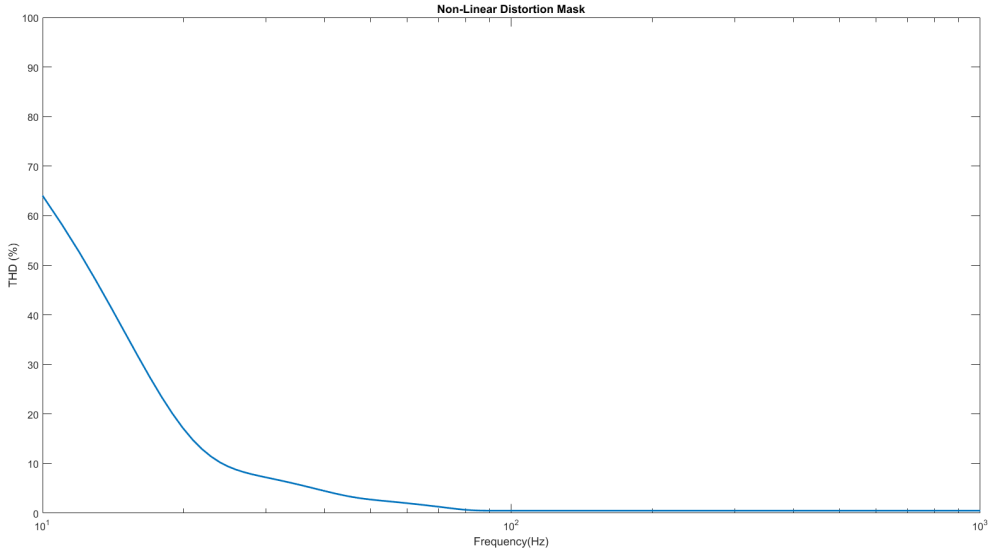


Figure A.1: The distortion mask as required by specification 1.3.2..

Bibliography

[1] NXP Semiconductors. (2002, November). *96 kHZ sampling 24-bit stereo audio ADC* [Online] Available: http://www.farnell.com/datasheets/809145.pdf

[2] ARM. *Cortex-A7 MPCore Technical Reference Manual Revision: r0p5* [Online]. Available: http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0464f/index.html

[3] J. A. Klaassen, S. H. de Koning *Bewegingstegenkoppeling bij luidsprekers*, Philips Technisch Tijdschrift, Jaargang 29, 1968

[4] Alan v. Oppenheim, *DT Filter Design: IIR Filters*, Massachusetts Institute of Technology, 2006

[5] Lennart Ljung, *Perspectives on system identification*, Annual Reviews in Control, 10 April 2010

[6] Karsten Øyen, *Compensation of Loudspeaker Nonlinearities - DSP implementation*, Norwegian University of Science and Technology, 2007

[7] Yaoyu Li, George T.-C. Chiu, *Control of Loudspeakers Using Disturbance-Observer-Type Velocity Estimation*, IEEE/ASME Transactions on Mechatronics, vol. 10, no. 1, February 2005

[8] H. W. Holdaway, *Design of Velocity-Feedback Transducer Systems for Stable Low-Frequency Behavior*, IEEE Transactions on Audio, September/October 1963 p. 155-173

[9] H. W. Holdaway, *Controlling the Upper-Frequency Characteristics of Velocity-Feedback Loudspeaker Systems*, IEEE Transactions on Audio, September/October 1963 p. 174-182

[10] Clark. J. Radcliffe, Sachin D. Gogate, *Velocity Feedback Compensation of Electromechanical Speakers for Acoustic Applications*, International Federation of Automatic Control, Triennial World Congress, June 30 - July 5 1996

[11] Karsten Wiedmann and Tobias Weber, *A grey-box modelling approach for the nonlinear parametric channel* IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP), 2014

[12] Khalid Mohammad Al-Ali, *Loudspeakers: Modeling and Control*, Graduate Division of the University of California at Berkeley, Fall 1999

[13] Michel Verhaegen, Vincent Verdult, *Filtering and System Identification: A Least Squares Approach*, Cambridge University Press, 2007

[14] Wilson J. Rugh, *Linear System Theory*, Department of Electrical and Computer Engineering, The Johns Hopkins University, 1996

[15] David Westwick, Michel Verhaegen, *Identifying MIMO Wiener systems using subspace model identification methods*, Signal Processing 52, p. 235-258, 1995

[16] Michel Werman, Daniel Keren, *A Bayesian Method for Fitting Parametric and Nonparametric Models to Noisy Data*, IEEE transactions on pattern analysis and machine intelligence, Vol. 23, No. 5, 2001

[17] Maurice Clerc, *Standard Particle Swarm Optimisation*, HAL archives ouverts, 2012

[18] Mojtaba Taherkhani, Reza Safabakhsh, *A novel stability-based adaptive inertia weight for particle swarm optimization*, Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran, 2015

[19] Xin-she Yang, *A New Metaheuristic Bat-Inspired Algorithm*, Department of Engineering, University of Cambridge, 2010

[20] Leon W. Couch II, *Digital and Analog Communication Systems*, Eighth edition, p171, 2013

[21] Jaco Salentijn, Oscar de Groot, *Amplifier Design in a Motional Feedback Audio System*, Department of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Delft, Netherlands, 2016

[22] Yann Rosema, Ramon Overwater *Digital Implementation of a Motional Feedback Audio System*, Department of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Delft, Netherlands, 2016