# Model Estimation & Quadrature-Point Controller Design

## For driving ultrasound

J.H.F.Jaspers & J.H.Metz

Delft University of Technology

TU Delft

# Model Estimation & Quadrature-Point Controller Design

## For driving ultrasound

by

J.H.F.Jaspers & J.H.Metz

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended publicly on Thursday June 29, 2023 at 01:30 PM.


| | | |
|---|---|---|
| Student number: | 5372925 | 5327253 |
| Project duration: | April 24, 2023 – June 30, 2023 | |
| Thesis committee: | dr. ir. A. J. van Genderen, | TU Delft, supervisor |
| | dr. ir. M. A. P. Pertijs, | TU Delft |
| | ir. A. J. M. Montagne, | TU Delft |
| | dr. ir. W. J. Venstra, | Quantified Sensor Technology |


*This thesis is confidential and cannot be made public until June 30, 2023.*



TU Delft

# Abstract

This report offers an in-depth analysis of the design and implementation of a subsystem within a system that will be driving a piezo-electric transducer. During operation, the position of the piezo-electric transducer is measured by means of interferometry. This is to achieve adaptive control which results in capturing time-varying dynamics of the piezo-electric transducer and the circuitry that is driving it.

The primary objective of the subsystem discussed in this report is to continuously estimate the transfer-response of the system using the position measured by the interferometer, in order to accomplish adapted model inverted control using a dynamically updating model. In addition, this subsystem design incorporates a controller that is crucial for obtaining undisturbed position measurements. This is achieved through a feedback loop that is controlling the DC operating point of the interferometer such that it is at its quadrature point.

# Preface

Thanks to Warner for proposing the project. Thanks to Arjan for supervising our project. And also to Michiel, Anton and Paul for helping one of the subgroups when they had questions. And last but not least, to all of our BAP group members: Marick, Nathan, Nic and Thomas!

*J.H.F.Jaspers & J.H.Metz*
*Delft, June 2023*

# Contents

# 1 Introduction

In this chapter the concepts that are fundamental for this project are introduced. Thereafter, the goal of this project is placed in this context. Finally, the document outline will be described.

## 1.1. Piezoelectric Transducers

The piezoelectric effect relates mechanical and electrical variables. Piezoelectric materials possess the capability to generate electric charge as result of an applied mechanical stress [1]. On the other hand, the inverse piezoelectric effect is the phenomenon in which the material undergoes mechanical deformation when an electric field is applied to the material. Where the piezoelectric effect is used for sensing applications, the inverse effect is used for actuation applications. This thesis is focused on the application of ultrasound piezoelectric transducers, which rely on the process of converting sound waves into electrical charge and vice versa by means of a vibrating piezoelectric material.

Ultrasound piezoelectric transducers are used in a wide range of applications. Often as a combination of sensing and actuation used in imaging techniques. Ultrasonic imaging is extensively used in the field of healthcare since it comes with the advantage of not having to perform a surgery and still be able to do diagnose patients [2, 3] . Pulse-echo techniques are used to inspect the quality of the structural integrity of materials in a non-destructive manner [4]. Other applications are in the field of (underwater) navigation and material cleaning [5, 6].

Characterization of piezoelectric transducers is essential when it comes to design of ultrasonic systems. Several tests are usually performed to obtain the electrical, mechanical and acoustic properties [7, 8]. The electrical properties are usually measured by performing an impedance test using an impedance analyser. Some important mechanical properties are the sensitivity and the frequency characteristics of the device. The sensitivity is usually obtained by quantifying the conversion of mechanical energy to electrical energy. For the frequency analysis of a ultrasonic transducer, several methods can be applied. One common method is to apply a frequency sweep on the device and measure the displacement using optical measurement techniques like interferometry or vibrometry [9]. Some acoustic properties are defined as beam-width and pressure level.

Additionally, it is important to characterize the nonlinear behaviour of ultrasound transducers. Especially for ultrasound transducers used for actuation. Driving ultrasound transducers at high power levels can result in the transducer operating outside of its linear regime, leading to non-linear behavior [10].

## 1.2. The goal of the project

As described, ultrasound transducers need proper characterization and can suffer from non-linear characteristics due to applying a high amount of power or due to temperature fluctuations. Currently the characterization of ultrasound transducers is performed for the region in which the transducer is going to be used. Characterization can be a time consuming job and requires a thorough understanding of the concept behind ultrasound transducers.

The goal of this project is to design a system that is able to drive an ultrasound transducer and achieving a flat transfer response within a given bandwidth, without performing a full characterization

of the ultrasound transducer. This will be achieved by using real-time position data of the ultrasound transducer. In order to capture the non-linear behaviour, it is essential to use real-time data other then position. Therefore, the aim is also to use data of a second feature for achieving a flat transfer response.

The data that is used for achieving a flat transfer response will thus be the measured position and a second feature that is possibly a source of non-linear behaviour. The position data should be measured such that it has no influence on the behaviour of the ultrasound transducer. Therefore, optical measurement techniques are most suitable. Common techniques that are used for measuring piezoelectric transducers are interferometry and vibrometry. For this project, interferometry is used for quantifying the displacement of the ultrasound transducer.

## 1.3. Thesis outline

The main subject of this thesis is the design of interferometer operating-point control system and a system that is processing the position data and estimating the ultrasound model. In chapter 2 the program of requirements for the whole system will be given. In chapter 3 a system overview will be described. In chapter 4 the interferometer setup will be introduced. In chapter 5 the design criteria for the control subsystem are discussed. After that, in chapter 6 the subsystem design will be discussed by dividing the subsystem in small functional sections. In chapter 7, the implementation of the designs discussed in chapter 6 are given with corresponding verification. Subsequently those techniques are implemented in a prototype and verified. Finally in chapter 8 the thesis results will be discussed, concluded and future works will be given.

# 2

# Program of Requirements

In this chapter the project of requirements for the project as a whole is described. The program of requirements is divided into mandatory and trade-off requirements. The mandatory requirements are divided in functional and non-functional requirements with the functional requirements being the requirements the system must do. And the non-functional requirements being the requirements representing the performance of the system.

## Mandatory Requirements

1. **Non-Functional requirements**
   (a) The steady state ripple of the Flatband must be within 3dB.
   (b) The bandwidth of the flatband must be 50kHz.
   (c) The bandwidth disturbance rejection must be at least 10kHz,
   (d) The operating center frequency of the signals is on the resonance point of the transducer, in the MHz range.
   (e) The sound power output must be 1W.
   (f) The costs of the prototype must be lower than 1500 EU.
   (g) The design should be realized using commercially available circuitry and equipment.
   (h) The complexity needed to operate the system is minimized
2. **Functional requirements**
   (a) The system must detect the vibrating position from the transducer using a non-contact high accuracy measurement device.
   (b) The system must control the position of the ultrasound transducer.
   (c) The time within the system should be functional is 120 seconds.
   (d) The system must be able to drive the ultrasound transducer with wave-forms composed of up to four frequency components.

## Trade-off Requirements

These are the requirements that should be adhered to as closely as possible where possible.

1. Minimize the steady state ripple of the Flatband.
2. Minimize the bandwidth disturbance rejection.
3. Minimize the costs of the prototype.
4. Minimize the time within the system should be functional.

# 3

# System Overview

For this project, three subgroups were formed. With the following main subjects divided into 3 parts: Interferometer operating point control & model estimation, data acquisition & system control and the ultrasound system. This thesis will describe the design process of the Interferometer operating point control & model estimation.

## 3.1. Interferometer Operating Point Control and Model Estimation

This part of the project has two subparts: Interferometer control and model estimation. The interferometer control has the task to make sure measurements done by the interferometer are not distorted by the shifting of the operating point. This is done by controlling the interferometer laser. The model estimation subpart should receive data from the data acquisition part and process this to update the model estimation. This model has to be transmitted to the system control part.

## 3.2. Position Measurement and System Control

Designed by subgroup [11]. The position measurement and system control part has 2 main tasks: control the ultrasound system according to an estimated model and measure the position of the piezoelectric transducer and transmit them to the Model estimation part.

## 3.3. Ultrasound system

Designed by subgroup [12]. This subgroup focuses on designing an amplifier circuit for a piezoelectric transducer to generate a flat frequency response in our frequency range of interest around 1 MHz. This part should create an amplifier stage that amplifies and converts the voltage signal into a current signal, which will then drive the piezoelectric transducer to create ultrasound.

# 4

# Introduction to the interferometry setup

In this chapter the basic concept of interferometry will be discussed followed by the introduction to the interferometry setup.

## 4.1. Position measurements using interferometry

Interferometry is a measurement technique that relies on the interference of waves. Waves contain an energy potential that is either positive or negative. When waves interfere, the total potential is influenced by both. For example when two waves of the same amplitude have the same phase at the same place and time, the total amplitude will be twice the amplitude of one wave. This is called constructive interference. But waves can also counter each other and thus reduce the size of the total amplitude. This is called destructive interference. The interference pattern that is formed by two waves can be used to measure distance. With the interferometry technique it is possible to reach nanometer accuracies [13]. For example: The formula for the wave of a laser beam with no loss is described as:

$$W(x,t) = Asin(kx - \omega t + \phi) \tag{4.1}$$

When adding extra phase due to the distance it travels further it is defined as:

$$W(x,t) = Asin(kx - \omega t + \phi + \frac{d}{\lambda} * 2\pi) \tag{4.2}$$

To measure the distance d, an interference pattern is needed that relies on d. An interference pattern can be created by using a partly permeable reflecting surface with another reflecting surface behind it. This will create an interference pattern in between the source and the partly permeable reflecting surface. The interference pattern for a semi(50%) permeable surface is:

$$W_i(x,t) = 0.5Asin(kx - \omega t + \phi) + 0.5Asin(kx - \omega t + \phi + \frac{2z}{\lambda} * 2\pi) \tag{4.3}$$

where z is the distance from the permeable surface to the full reflecting surface behind it. Due to the fact that it has to travel back and forth, the distance z must be multiplied with a factor two.

Rewriting this with the sine-cosine relations gives:

$$W_i(x,t) = Acos(\frac{2\pi z}{\lambda}) * sin(kx - \omega t + \phi + \frac{2\pi z}{\lambda}) \tag{4.4}$$

The normalized average power of the interference pattern is:

$$\frac{1}{T}\int_0^T W_i^2\, dt \tag{4.5}$$

The power of the interference pattern is then expressed as:

$$0.25 * A^2(1 + cos(2\pi\frac{2z}{\lambda}) = A_i(1 + cos(2\pi\frac{2z}{\lambda}) \tag{4.6}$$

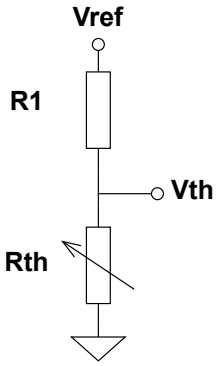



Figure 4.1: Voltage divider with a thermistor

The intensity of light is defined as:

$$I = P/S \tag{4.7}$$

where S is the area and P the power on the area.

With a light detector the intensity can be measured and translated into a displacement.

## 4.2. Controlling the wavelength of a laser diode

The wavelength of a laser diode is dependent on the temperature of the diode[14, 15]. To be able to control the wavelength with a voltage controlled signal, the temperature of the laser diode should be adjusted. A solution for obtaining a certain temperature out of a voltage is the use of a thermistor. This can be done by placing a thermistor in a voltage divider circuit 4.1. The divided voltage is then dependent on the temperature, but also the other way around. The resistance level of the thermistor can be changed by applying a different voltage at the Vth node. The thermistor will then obtain a temperature depended on the resistance level. This relation is described in the Steinhart-Hart equation:

$$\frac{1}{T} = A + B * \ln R + C * (\ln R)^3 \tag{4.8}$$

Where T is the temperature in Kelvin, R is the resistance in Ohms and A, B and C are the parameters of the thermistor used.

## 4.3. The setup

The setup provided for this project contains a laser module designed by Quantified sensor technology. The laser module has one input: DC temperature set point. With this input it is possible to heat or cool the laser diode and thus change it's wavelength as described in section 4.2. The voltage divider parameters are described in Table 4.1 The laser module has three outputs: an optical, a DC and an AC output. The optical output is connected with a laser diode inside the module. This laser diode sends a laser beam into the optical output. For the setup this output is connected with an optical fiber that is placed in a steady place right in front of a piezo electric transducer. The setup in the lab can be seen in Figure 4.4, but may be hard to interpret. A schematic image of the setup can be seen in Figure 4.3. The partly permeable surface in this setup is the end of the optical fiber core, because the laser beam cannot exit the fiber fully. The reflecting surface in this setup is a reflecting piece of tape that is attached to a piezo electric transducer. The distance in between the end of the optical fiber and the piezo electric transducer can be adjusted by moving the piezo electric transducer nearer or further away from the optical fiber. In the module a light detector measures the intensity of the interference pattern that comes back and converts it into a voltage signal. The AC output gives this voltage signal high pass filtered above 20KHz. This output can be used to measure harmonics of the piezo electric transducer higher that 20 KHz. The DC output gives the voltage signal low pass filtered. This DC output signal is used to measure the temperature of the laser.

The wavelength measurements dependent on the temperature are provided by Warner of Quantified Sensor Technology. They are shown in Figure 4.5

| | |
|---|---|
| Vref | 1.5V |
| R1 | 10K Ω |

Table 4.1: Parameters of the voltage divider

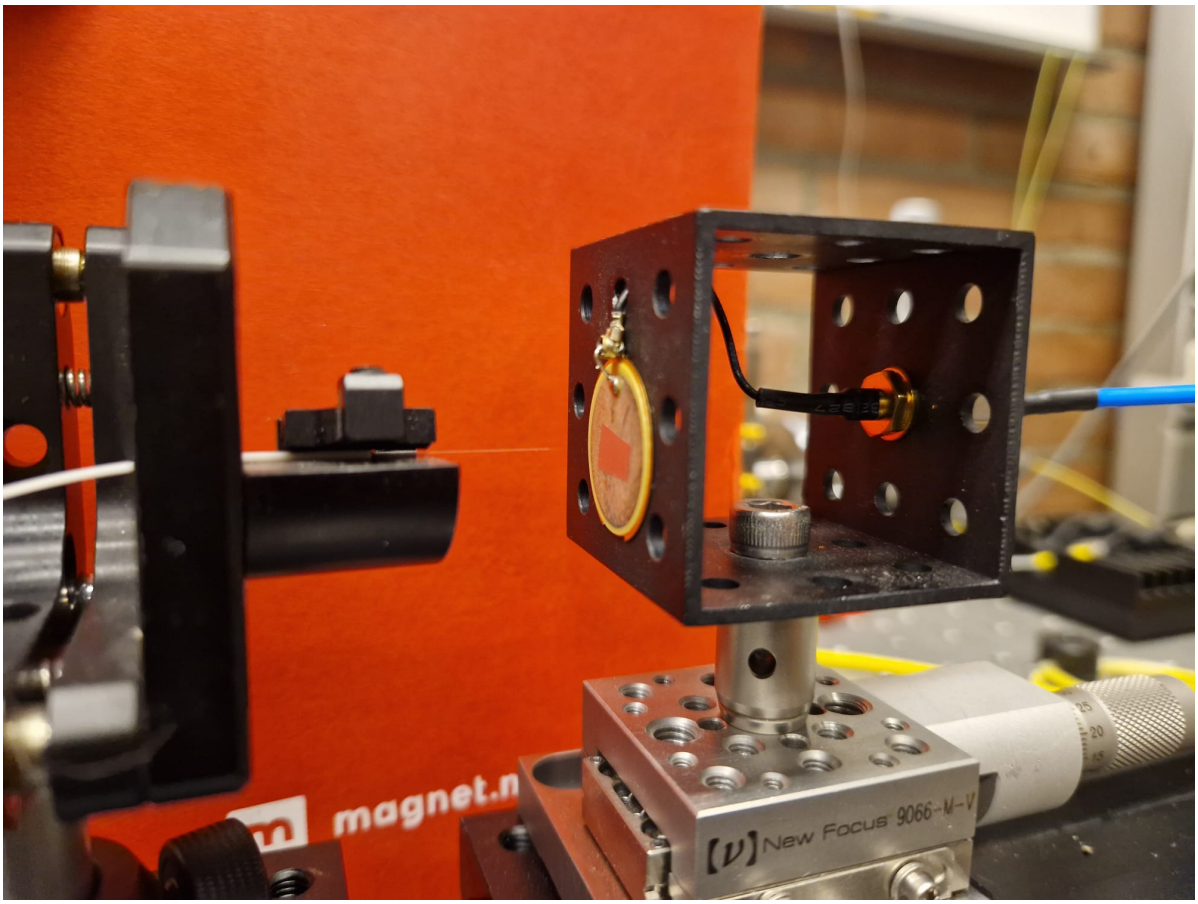



Figure 4.2: Interferometer setup in Quantified Lab

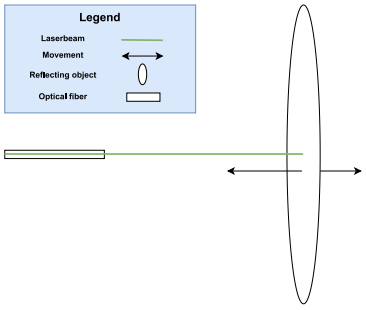



Figure 4.3: Schematical interferometer setup with an optical fiber and moving surface, laserbeam is projected in green but has a non visible wavelength

| Port | DC output | AC output |
|---|---|---|
| Frequency | 0 - 20Khz | 20Khz and above |

Table 4.2: AC & DC output ports of the laser module, with their frequency range

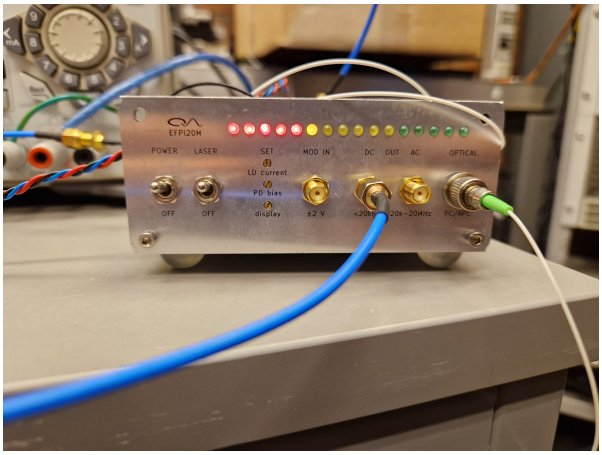



Figure 4.4: Laser Module in and outputs

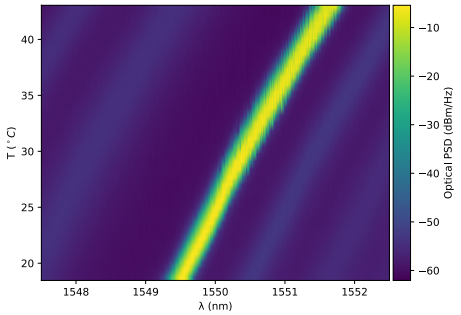



Figure 4.5: Optical power spectral density dependent on temperature and wavelength. Measurements provided by Warner of Quantified Sensor technology.

# 5

# Design Criteria for the Model Estimation subsystem

In this chapter the design limitations and criteria for this subsystem will be discussed. The design limitations are the result of physical limitations caused by design choices within the overall project. These limitations define the design criteria for this subsystem which results in a subsystem specific program of requirements.

## 5.1. Design Limitations for the Model Estimation subsystem

### Sample rate due to high sample resolution

As described in section 1.2, interferometry is used for measuring the position of the ultrasound transducer. The sample-rate at which the interferometer is sampled is defined by the ADC, which is 65Msamples/second. This results in a sample period of 15.38 ns/sample which is assumed to be too short for applying a system that is controlled by a discrete feedback loop. This limitation defines the functional design of this subsystem, namely model estimation for adaptive control by inversion of the ultrasound-system magnitude response.

### Bin size of position data

Since the time for the system to be functional is limited, the bin size is limited. Ideally, for model estimation it is preferred to have as much as possible data to process. So, a balance of bin size and run-time is required such that bot requirements are met. More in depth analysis is found in section 6.2.

### Feed-forward implementation size

The position measurement and system control is partly implemented on a FPGA. The requirements for that subsystem leads to a design limitation for this subsystem. Due to the limited implementation size for the FPGA, it is required to choose a model that has acceptable implementation costs for the FPGA.

### Feature number range FPGA

The features that will be used to define a model for the ultrasound system are implemented on the FPGA as 16 bit floating points. This limits the size range of the feature numbers. The upper limit is $abs(F \cdot P_1) < \sqrt[8]{2^{16}} = 4$. And the lower limit is $abs(F \cdot P) > \sqrt[8]{2^{-14}} = 2^{-\frac{14}{8}} \approx 0.3$.

### Interferometer operating point

The measurement technique introduces a source of signal distortion if no proper operating point control is designed and implemented. This could limit the quality of the measurement results and requires special attention. The operating point of the optical interferometer should be as close as possible to the quadrature-point such that the signal distortion is limited. This concept will be discussed in more detail in section 6.3.

## 5.2. Program of Requirements for the Model Estimation subsystem

Based on the design limitations for this subsystem that are raised by design choices made for the overall system and the other subsystems and based on the overall program of requirements, a subsystem specific program of requirements is composed. The requirements are divided into mandatory and trade-off requirements. The mandatory requirements are subdivided in non-functional and functional requirements.

### Mandatory requirements

1. **Non-functional requirements**
   (a) The maximum contribution to the steady state ripple of the flatband is 0.5dB.
   (b) The bandwidth of the flatband must be at least 100kHz.
   (c) The operating center frequency of the signals is on the resonance point of the transducer, in the MHz range
   (d) The maximum contribution to the cost of the prototype is 50 euro
   (e) The phase error of the operating point with respect to the quadrature point must be lower then 0.1 radians.
2. **Functional requirements**
   (a) The time within the system should be functional is 120 seconds.
   (b) The subsystem must provide an user interface.
   (c) The number of frequency components of the waveform is at least 4.
   (d) For each frequency component, an amplitude must be asked as input from the user.
   (e) The subsystem must be able to accept the position-data bin received from the subsystem Position Measurement and System Control
   (f) The operating point must be controlled.
   (g) Provide the subsystem Position Measurement and System Control with parameters that are required for signal generation.
   (h) Communication lines with the FPGA should be established.

The requirements 1a, 1b, 1c, 1d, 2a and 2c are defined by the overall system program of requirements. The other requirements are based on the specific design limitations for this subsystem.

### Trade-off requirements

1. Minimize the contribution to the steady state ripple of the Flatband.
2. Minimize the contribution to the cost of the prototype.
3. Minimize the maximum steady-state signal distortion as result of operating-point control.
4. Minimize the time within the system should be functional.
5. Design the subsystem with the principle of flexibility, such that second feature can be added to the design easily and feature choice is flexible.
6. Provide characterization data as product.
7. Provide a graphical user interface.

# 6

# Subsystem Design

## 6.1. Model Estimation

In order to achieve a flat transfer response, the ultrasound system needs to be controlled which is done by controlling the position of the ultrasound transducer. As described in section 5.1, the system as a whole is limited to control the ultrasound transducer by means of online feed-forwarding or offline feed-forwarding (by characterizing the ultrasound transducer extensively). Feed-back in the sense of real-time measurements is not possible due to the sample time. As described, characterization of ultrasound transducer requires a thorough understanding. In addition, the system is not flexible for driving different ultrasound transducers when control by characterization is performed

The ultrasound control system is designed as an adaptive inverse feed-forward controller. This is a controller that is using an estimation of the magnitude response of the ultrasound system to determine the control signal that is required to drive it with a flat response. This subsystem is responsible for estimating the model parameters that are estimating the transfer response of the ultrasound system denoted by $|H(f)|$ in Figure 6.1.

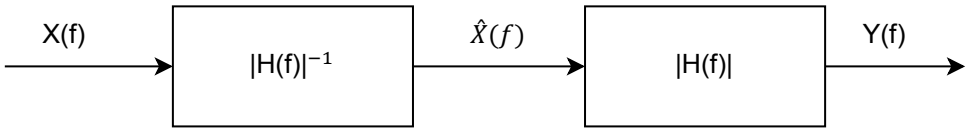



Figure 6.1: Inverted adaptive control scheme, with $|H(f)|$ being the ultrasound system

The magnitude response of a system is defined as the ratio of the magnitudes of the output and input of a system:

$$|H(f)| = \frac{|Y(f)|}{|X(f)|} \tag{6.1}$$

Since it is expressed as a ratio, the magnitude response also can be expressed as ratio of power:

$$|H(f)| = \frac{P_Y(f)}{P_X(f)} \tag{6.2}$$

In order to estimate the transfer function $|H(f)|$ in an inverted feed-forward scheme as illustrated in Figure 6.1, the input power of the signal $\hat{X}(f)$ is the true system input signal. Since this signal is generated internally, the input power is derived from that generated signal. On the other hand, the output power is derived by processing the output that is obtained by measuring the position of the ultrasound transducer. The processing of the output (position) data is done in the frequency domain and the process will be discussed in section 6.2.

Before moving to the derivation of the input power, the model estimation method is required because the ultrasound system input signal is generated using this model. The model estimation method is designed such that it meets the requirement for implementation costs for the subsystem Position

Measurement and System Control. Therefore, the transfer response will be estimated by a magnitude response optimized on a polynomial. In order to take into account the nonlinear behaviour of the ultrasound system, the magnitude response will not only be dependent on the frequency but also on a second variable. The variable that will be used for estimating the non-linear behaviour is required to cause non-linearity. This leads to two obvious possible choices of the second variable, namely temperature of the ultrasound transducer or the total input power of the ultrasound system. The system will be designed such that the implementation of the second variable will be flexible. This flexibility provides an opportunity to test and analyse the influence of different parameters on the non-linear behaviour.

The model thus consists of two features. The first feature being the frequency (f) and the second feature being the extra feature (z). This will change Equation 6.1 to:

$$|H(f,z)| = \frac{|Y(f,z)|}{|X(f,z)|} \tag{6.3}$$

Both features are an array that represents a polynomial to the seventh order.

The data that will be used for optimization is obtained by collecting ultrasound system output power and input power.

$$\vec{\theta}(f) = \begin{bmatrix} 1 \\ f \\ f^2 \\ f^3 \\ f^4 \\ f^5 \\ f^6 \\ f^7 \end{bmatrix} \qquad \vec{\phi}(z) = \begin{bmatrix} 1 \\ z \\ z^2 \\ z^3 \\ z^4 \\ z^5 \\ z^6 \\ z^7 \end{bmatrix} \tag{6.4}$$

In order to optimize the least-squares for those two arrays, a weight array is defined. This weight array consist of a weight for every combination of the two feature vectors. The magnitude estimation is then defined as:

$$|\hat{H}(f,z)| = [\vec{\omega} \cdot \vec{\theta}(f)] \cdot \vec{\phi}(z) \tag{6.5}$$

Where $\vec{\omega}$ is a row vector representing all the weights that will be optimized.

For optimization, the magnitude samples are required. As discussed, for calculating the magnitude the ultrasound system input power and output power are required. The input power is calculated by filling in Equation 6.5. This gives the magnitude corresponding to f and z. When the input and output powers are derived, the magnitude corresponding to f and z can be calculated and stored.

Note that initially no weights are defined. So, before starting the optimization process, an initial model is required. This could be a randomly picked model or a rough estimation of the model.

### 6.1.1. Calibration: frequency sweep

Since the user input is only composed of four frequency components, the magnitude samples that are collected are concentrated at these frequencies. When only the user input is used as magnitude samples, the model will eventually be biased at certain frequencies, meaning that the data can be fitted incorrectly. In order to prevent this, a wide range of data should be collected over the whole bandwidth. This can be done by implementing a calibration routine that is sweeping the input of the ultrasound system with frequencies over the bandwidth. The frequency sweep will be performed at start up of the system. It could also be performed during operation, but that is up to the user.

## 6.2. Data Processing

The subsystem Position Measurement and System Control is collecting data which is equivalent to windowing a time-signal with a square window. In order to estimate the magnitude response of the system that is controlled, proper frequency data for the input and output of the system is required. This will be achieved by processing the windowed time-domain position data by first applying a Hann-window, then performing a FFT and finally selecting the samples containing all the power of the given frequency component.

### 6.2.1. Spectral leakage reduction

When performing a FFT on a finite length time-domain signal, spectral leakage occurs. By definition, a FFT is performed over a finite length signal. This results in a frequency resolution at which frequencies can be represented. If the frequency component that is present in the time-domain signal is not a multiple of the frequency resolution, the power of this frequency component is smeared over neighbouring samples. In other words, if the frequency does not correspond to a frequency bin, the power of that frequency component will be leaked. For estimating the magnitude response of the system, it is required to obtain a precise estimation of the systems output power. Therefore, the spectral leakage problem needs to be resolved before calculating the magnitude for a single frequency component.

There are several possible approaches to consider for the spectral leakage compensation. By choosing the window-size such that the frequency component always fits in a frequency bin, there will be no spectral leakage. So, by varying the window-size, the problem can be easily solved. However, this forces the subsystem Position Measurement and System Control to implement a variable window-size depending on the frequency components and since a maximum of four frequency components present in the wave-form is allowed, capturing a period of such a periodic signal could lead to large window-sizes. A more robust approach is to choose a constant window-size and apply another window to the square window. By multiplying the time-domain signal with a window-function, the spectral leakage can be reduced. Multiplication in the time-domain is equivalent to convolution in the frequency-domain. By choosing a window that has a narrow shape in the frequency domain, it is possible to detect the frequency components and reduce spectral leakage.

There are several window-types developed, each for a specific signal type. From the overall program of requirements the signal type can be derived. The signal type is a sinusoidal wave composed of four different frequency components, that are not specifically required to have frequencies that are close to each other. For this application, the Hann-window is chosen. The Hann-window is appropriate for signals composed of multiple sine waveforms [16] and is defined as:

$$w(n) = 0.5 - 0.5cos(\frac{2\pi n}{M-1}) \qquad 0 \leq b \leq M-1 \tag{6.6}$$

When windowing the signal, a fraction of the total signal power will be lost. This needs to be compensated for in the frequency domain. The power loss appears to be a constant value, for the Hann-window this is 1.63 [17].

### 6.2.2. Minimal required window size

The window size is determined by several requirements. The minimal required window size is determined by the maximal required frequency resolution. The window-size is related to the frequency resolution and the sample-rate as follows:

$$N = \frac{F_s}{2 \cdot \Delta f} \tag{6.7}$$

(6.8)

With $N$ being the window-size in samples, $F_s$ being the sample-rate in Hertz and $\Delta f$ being the frequency resolution in Hertz- per bin. The maximal required frequency resolution is not a system requirement. However, in order to implement the frequency sweep time-efficiently, a maximum frequency resolution is to be determined. As discussed in subsection 6.1.1, during one sweep iteration, four frequency components are the input of the system. The frequency components are spaced uniformly over the bandwidth such that the frequency sweep duration is reduced to four times smaller compared to a frequency sweep that only sets one frequency component on the input of the system per iteration. The required frequency resolution depends on the required distance between the frequency components.

The distance between the frequency components can be expressed in terms of Hertz and in terms of bins. In order to express it in terms of bins, a minimal required distance between bins is derived. This is done by simulation of discrete sine-waves swept over the systems requir ed bandwidth and also swept over a range of window-sizes that are all a powers of two. Consequently, for each generated sine-wave the number of samples that contains 90% of the total power present in the signal will be extracted. The results are plotted in a color-map in Figure 6.2

The results of the simulation show that the number of bins that contain approximately all the signal power is always between three and five. In other words, for these ranges of frequency and window-size

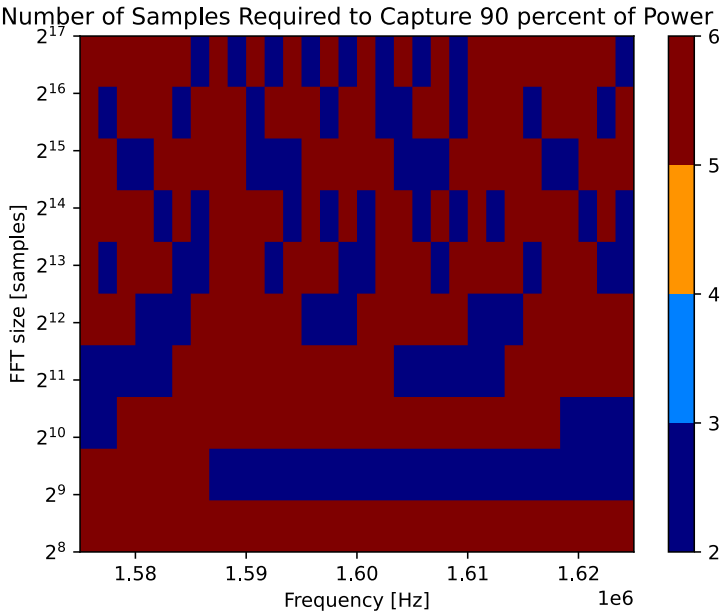



Figure 6.2: Approximation of the number of samples that contain 90% of the signal power by varying the frequency and window-size

the minimal distance between two frequency components in terms of bins is six. Now the minimal distances between frequency components are determined in both terms of bins and Hertz, the frequency resolution can be determined. The frequency resolution is the distance in Hertz per bin.

$$\Delta f = \frac{\Delta B}{\Delta N} = \frac{(\frac{50kHz}{4})}{6} \approx 2.08 \qquad [\frac{kHz}{bin}] \tag{6.9}$$

With $\Delta B$ being the distance between frequency components in terms of Hertz and $\Delta N$ the distance between frequency components in terms of samples. This resolution determines minimal window-size that is required for improving the frequency-sweep efficiency by a factor of four.

$$N_{min} = \frac{F_s}{2 \cdot \Delta f} = \frac{65 \cdot 10^6}{2 \cdot 2.08 \cdot 10^3} \approx 15.6 \cdot 10^3 \qquad [samples] \tag{6.10}$$

### 6.2.3. Window-size upper limit

The upper limit of the window-size is directly determined by the allowed compute time. Ideally, the upper limit of the window-size is infinite, since that gives the highest accuracy. Obviously, an infinite window-size is not possible and there is also limit caused by implementation size. However, the implementations size is less likely to be the limit compared to the compute time. So, the limit will be determined by the compute time and that is tuned at the integration phase of this project. Note that the FFT is most efficient if the FFT-size, and thus window-size, is a power of 2.

### 6.2.4. Signal to noise compensation

Given the type of signal, a sinusoidal wave composed of at maximum four frequency components, the signal to noise ratio (SNR) can be improved in the frequency domain if the spectral-leakage allows it. By looking at the result of Figure 6.2, Parseval's Theorem can be used to determine the total output power of a frequency component. Note that it is assumed that the minimal distance between frequency components is such that no spectral power is overlapping between two frequency components. So, by assuming that all the spectral power of a frequency component is within six bins and no overlap in spectral power of separate frequency components is possible, Parseval's Theorem for discrete signals

can be applied to obtain the total power of a single frequency component:

$$\sum_{n=0}^{N-1} |x[n]|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2 \qquad (Parseval's \quad Theorem) \tag{6.11}$$

Parseval's Theorem states that the energy is conserved when applying a DTFT to a time signal. This can be used to determine the power of a single frequency component that is present in the time signal. The power for each frequency component present in the signal is performed by first calculating the frequency bin number (Equation 6.12) in which the original (user input) frequency is placed. Consequently, the power corresponding to the frequency component is approximated by Equation 6.13.

$$\#N = round(\frac{f}{\Delta f}) \tag{6.12}$$

$$P_f = \sum_{\#N-\frac{\Delta N}{2}}^{\#N+\frac{\Delta N}{2}} |X[k]|^2 \tag{6.13}$$

Note that Equation 6.13 is true only if the power spectrum is normalized to the samples in the right half of the spectrum.

Comparing the SNR before extracting the power ($SNR_{original}$) to the SNR after extracting the power ($SNR_{processed}$) provides insight of the quality of the process discussed in this section and it provides insight in the maximum noise floor that is acceptable. The $SNR_{original}$ is thus defined by the total power present in the signal an the total noise that is present over the bandwidth. The total noise is assumed to be white noise that is defined by the noise floor. The SNR before dataprocessing is defined as:

$$SNR_{original} = \frac{P_{signal}}{NoiseFloor \cdot B} \tag{6.14}$$

The dataprocessing reduces the total noise. The total noise after data processing is defined as the product of the frequency bins containing all the power, as derived in subsection 6.2.2, and the noise per frequency-bin. Consequently, the noise per frequency-bin is the total noise divided by the window-size.

$$TotalNoiseReduced = \Delta N \cdot NoisePerBin \tag{6.15}$$

$$NoisePerBin = \frac{TotalNoise}{N} = \frac{NoiseFloor \cdot B}{N} \tag{6.16}$$

By combining Equation 6.15 and Equation 6.16, the reduced noise is derived which is used to calculate $SNR_{processed}$

$$TotalNoiseReduced = \frac{\Delta N \cdot NoiseFloor \cdot B}{N} \tag{6.17}$$

$$SNR_{processed} = \frac{P_{signal}}{TotalNoiseReduced} = \frac{P_{signal} \cdot N}{\Delta N \cdot NoiseFloor \cdot B} \tag{6.18}$$

From this equation can be seen that if the window-size increases, the SNR also increases. This is due to the fact that if the window-size increases, the total noise present, is divided over more bins. Also, if the noise-floor decreases, a lower minimum signal power that is allowed. This relation shows clearly the dependencies of different variables that are a result of design choices. It is also important to realize that the noise-floor is a result of the design of the subsystem Position Measurement and System Control.

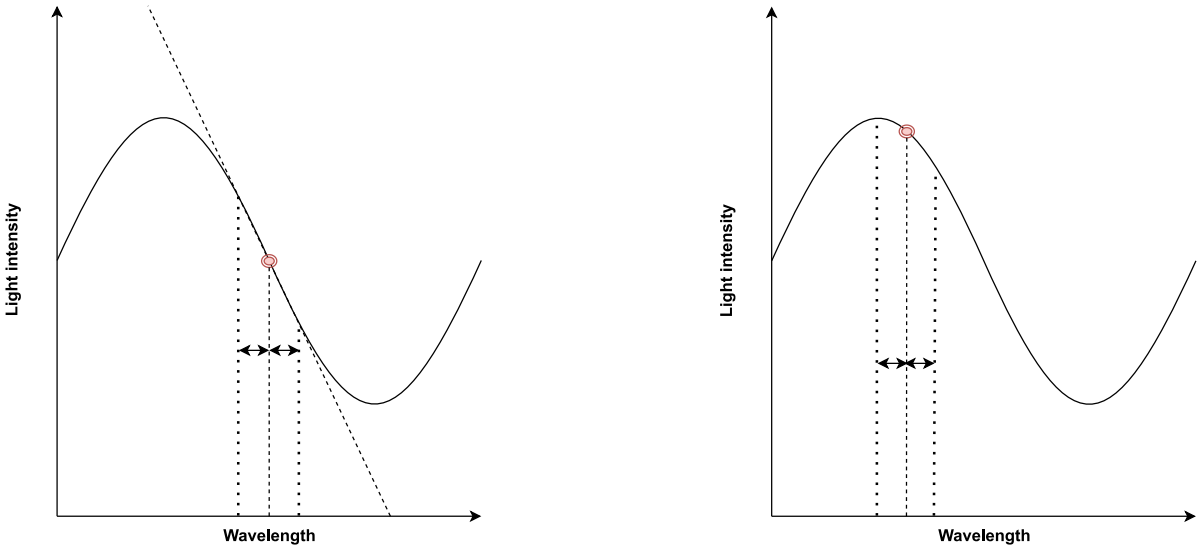
## 6.3. Interferometer operating-point control

The goal of the system is to measure the displacement of the piezo electric transducer. To measure a displacement an operating point should be chosen. This operating point cannot be chosen randomly. The phase difference is measured by measuring the voltage level of the AC output for frequencies higher than 20 KHz or the DC output for frequencies lower than 20 KHz. For measuring the phase difference, the linearity of a sine or cosine wave around y = 0 is used. In this way the phase difference can be linearly approximated. To be able to measure the phase difference correctly, the operating point should be at a quadrature point of the interference pattern as in 6.3a. When the operating point is wrongly chosen, such as in 6.3b, the derivative of the light intensity is much smaller when going up in the curve than when it is going down the curve. This would result in a bigger or smaller phase difference measured than it actually is, depending on if the curve is ascending or descending. In this way a proper linear approximation cannot be made anymore. This means that when using the Equation 4.6 the wavelength should be chosen that the operating point lies at a quadrature point of the interference wave.

$$cos(2\pi\frac{2z}{\lambda}) = 0 \tag{6.19}$$

and thus:

$$z = \frac{k + 0.5}{4} * \lambda \tag{6.20}$$




(a) Quadrature point and range

(b) Non ideal operating point and range

Figure 6.3: Quadrature point and non-ideal operating point

### 6.3.1. Operating point control

Different techniques are developed for controlling the operating point. To design according to the requirements, simple to implement and financially cost effective methods are investigated. A common method is to implement the control system as a feedback-loop [18, 19, 20]. The interference pattern is low-pas filtered which is also the case for the interferometer that will be used for this project. The method relies on controlling the wavelength by measuring the dc level of the interference pattern. However, a feedback loop requires a setpoint that it will track. The setpoint determines the operating point of the interferometer and should be the quadrature point in order to meet the requirement of 0.1 phase error. This feed-back loop will be implemented as a PID-controller because of the simplicty, robustness and cost effectiveness. In addition, the quadrature point also needs to be found as discussed in the next subsection, which means that combining both on one component would be usefull.

### 6.3.2. Quadrature point estimation by finding the inflection point

There are several methods to derive the optimal operating point. The different methods rely on assumptions made and need to be verified afterwards using measurement data.

**Extreme value detection**

For the first method it is assumed that the wave-form approximates a sine-wave with constant amplitude and constant offset. For a single frequency sine-wave, the quadrature point is always at its equilibrium. The equilibrium can be calculated by taking the average of the maximum value and the minimum value. To make this method more robust against errors that influence the calculated equilibrium, some sort of outlier detection system should be implemented that is able to detect outliers that are either too high or too low. Z-score, MAD and boxplot methods are considered from which MAD is the most effective method [21].

To detect outliers the Median Absolute Deviation threshold can be used.

$$MAD = median(|X_i - median(X)|) \tag{6.21}$$

And with the MAD outliers can be detected with a threshold:

$$C = \frac{|X_i - median(X)|}{MAD} > Threshold \tag{6.22}$$

If the C value is bigger than the threshold, it will be detected as an outlier.

**Offset filtering**

Another approach is based on the assumption that the wave-form can be approximated as a sine-wave with an x-axis dependent offset. By high-pass filtering the wave-form, the DC level and thus the offset is filtered out of the signal resulting in a sine-wave without offset. The quadrature points are then located at the zero-crossings. By storing the indices of the zero-crossings, the dc values can be extracted from the original wave-form. This results in a set of possible setpoints. Note that this method can be used combined with the first method.

**Least-square fitting**

The quadrature point is located where the second derivative is equal to zero, which is the definition of an inflection point. This approach is to find the inflection point by optimizing an analytical function and finding the second derivative. This function can be found by performing a least-squares fit. However, the waveform is not a pure sine-wave. The offset and the amplitude can be varying with an increase of the x-axis. This depends on the measurement setup and other complex influences. This prototype function is rather complex, since it is depending on multiple variables, it is prone to optimize to the wrong minimum of the cost-function and thus prone to unreliable results. In order to prevent this optimisation problem, the optimisation variables could be estimated beforehand. For example, the frequency can be estimated by performing a FFT on the measurements and extracting the most dominant frequency component. It is preferred to estimate every optimisation variable, thus make a good initial guess, before performing the optimisation algorithm in order to achieve a correct analytical estimation.

**Numerically slope estimation**

The second approach is to find the inflection point by calculating for all samples the slope with respect to neighbouring samples. By simply finding the absolute maximum slopes (above a threshold), the inflection points can be derived. The downside of this approach is that it is sensitive to noise. This can be resolved by creating clusters of the absolute slopes above the threshold. This will reject large slopes that are caused by outliers. From the clusters of large slopes, the center sample can be found that will represent the inflection point locally.

**Quadrature point estimation: Trade-offs**

To choose which method is best for this application, in Table 6.1 the trade-offs are presented. The criteria are decided to be implementation simplicity, system interruption and error resistance. The estimated effort it would take to achieve a functional version of the method is defined as implementation simplicity. System interruption is defined as the way it has to interrupt the flow of the overall system

by performing the quadrature point estimation. Error resistance is defined as how well the system will resist errors.

Note that in chapter 7 the implementation of the overall system and subsystem is discussed which influences the trade-offs. The implementation simplicity is due to the possibility to perform methods on a PC. However, if a method should be implemented on the PC, it is interrupting the flow of the overall system. It is decided that the extreme value method and the slope calculation can be implemented on the microcontroller such that it is isolated from the system and thus is not interrupting the flow of the overall system. The offset filtering and least-square methods are decided to be implemented on the PC as a Python script, since it is likely that it needs large calculations. Due to implementation in Python it means that it is relatively simple to implement but on the other hand is it interrupting the system.

Table 6.1: Trade-off table for quadrature point estimation methods

| Trade-off criterion | Extreme value | Offset filtering | Least-square | Slope calculation |
|---|---|---|---|---|
| Simplicity | - | ++ | - | - |
| Interruption | ++ | - | - | ++ |
| Error Resistance | - | - | ++ | -+ |

# 7

# Implementation and Verification

In this chapter an overview of the hardware implementation of the overall system from the perspective of this subsystem will be given followed by the hardware implementation of this subsystem. Consequently, the implementation of the theoretical solutions discussed in chapter 6 will be discussed and verified. Also, problems that raises due to chosen hardware implementation will be discussed.

## 7.1. Hardware implementation

The overall system can be divided into six hardware components namely ultrasound transducer, ultrasound driver circuit, FPGA, interferometer, microcontroller (MCU) and PC.

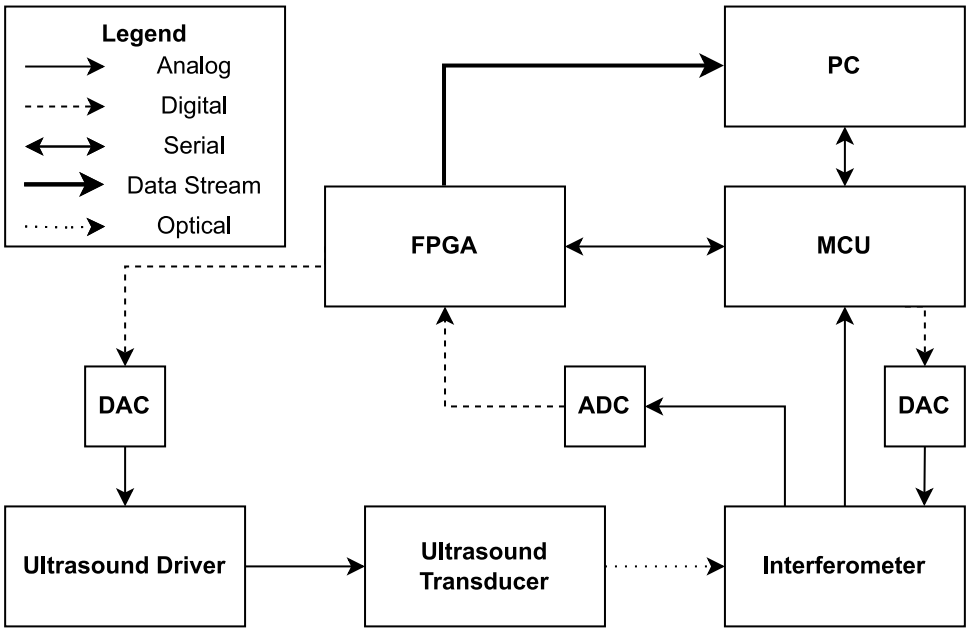



Figure 7.1: Overall system hardware implementation overview

Figure 7.1 illustrates the overall system hardware implementation in which the most important connections are illustrated. The legend shows the connection types. The figure is divided into two parts, the top half of the figure contains all digital components and the bottom half contains all analog components. The analog and digital domain is connected via analog to digital converters (ADC) and digital to analog converters (DAC). Note that the analog connection from the interferometer to the MCU is connected to an internal ADC of the MCU. This connection is representing the DC-component of the interference signal and is established by an internal low-pass filter of the interferometer. The AC-component of the interference signal is connected via an ADC to the FPGA which will be representing the position of

the ultrasound transducer during operation. The connection of the MCU to the interferometer is connected to the internal wavelength-controller which will be used for controlling the operating-point which is discussed be discussed in item 1e.

As discussed earlier, the functions are divided in three categories. Namely ultrasound, data acquisition & system control and model estimation & interferometer operating-point control. The ultrasound is implemented as the ultrasound driver that is amplifying a signal and thus driving the ultrasound transducer to produce ultrasound. Secondly, the data acquisition & system control is implemented on a FPGA that will be collecting the transducers position data and forwarding it to the PC via the datastream. The FPGA is also generating the signal that is driving the ultrasound transducer. This is done by inverting the model and generating the required signal to have a flat response on the transducer. These model parameters are estimated by the PC and forwarded via the MCU to the PC. The second task for the PC and MCU is derive the quadrature point of the interferometer and controlling the operating point such that it stays at the ideal operating point.

The motivation for the hardware implementation of the ultrasound and position measurement & system control can be found in [11, 12]. For this subsystem, the main objectives are providing a model estimation and controlling the operating-point of the interferometer. In order to effectively address these objectives, they are further divided into specific problems that defined the features which are needed to be designed and implemented. Those features are categorized into two groups: features that require fast and uninterrupted execution, and those that are computationally expensive. To meet these two global requirements, it is chosen to implement the subsystem mainly on a PC and partly on a microcontroller. A more elaborate explanation is given in the dedicated sections.

## 7.2. PC

As discussed, the subsystem objectives results in two types of implementation needs. To meet those requirements, a part of the subsystem is implemented as a Python module. Python offers a wide range of capabilities and features and thus it is a good medium for implementing a system that faces various distinctive problems. The functions that are implemented in the Python module are: the user interface, data processing, model estimationand quadrature-point estimation. This section will be used to describe how these functions are implemented and validated.

### 7.2.1. Implementation overview

In order to successfully implement CPU-bound tasks and IO-bound tasks in one Python module, the tasks that need to be performed are categorized as CPU-bound (or compute-bound) tasks and IO-bound (InOut-bound) tasks. CPU-bound tasks are limited by the processor, the tasks are performing large sequential calculations. On the other hand, IO-bound tasks are limited by in- and outputs from for example networks, data from memory and user inputs. Therefore, the module is implemented in a multi-processing approach. As the term suggests, multi-processing is assigning tasks to a processor and its memory space. In other words, processes that can be executed independently are implemented as a process, such that all processes are executed on separate processor cores. The processes execute in parallel and thus the tasks are performed in parallel of each other. This improves the run-time of the program, especially for paralyzing the CPU-bound tasks. However, IO-bound tasks can still be blocking the processes. This is resolved by implementing each individual process as multiple threads. Multi-threading is different from multi-processing. Threads are not executed sequentially as default programs do, but threads are alternatingly given run-time. So, if a IO-bound task is blocking a sequential task, the delay that will be caused is reduced since the threads will be taking turns.

The module consists of five processes that will be sharing three global memory addresses. Memory that is shared between processes should be synchronized to prevent racing conditions. Data acquisition from shared memory is implemented using data locks. When memory is accessed, it will be locked so that when another process tries to access it, it is blocking until the memory is unlocked. The other inter-process data transfer is implemented as so called Pipes. A pipe is an one way data transfer between processes from which one is the transmitter node and the other is the receiver node. This method has a built in locking mechanism. The five processes with the shared memory are illustrated in Figure 7.2. Inter-thread communication is not illustrated. However, each thread is able to access the local memory. Note that time in terms of sequences is denoted as going downwards in the illustration. However, the sequences are cyclic, the program will not end if it reached the bottom. Also, each process is running

independently.

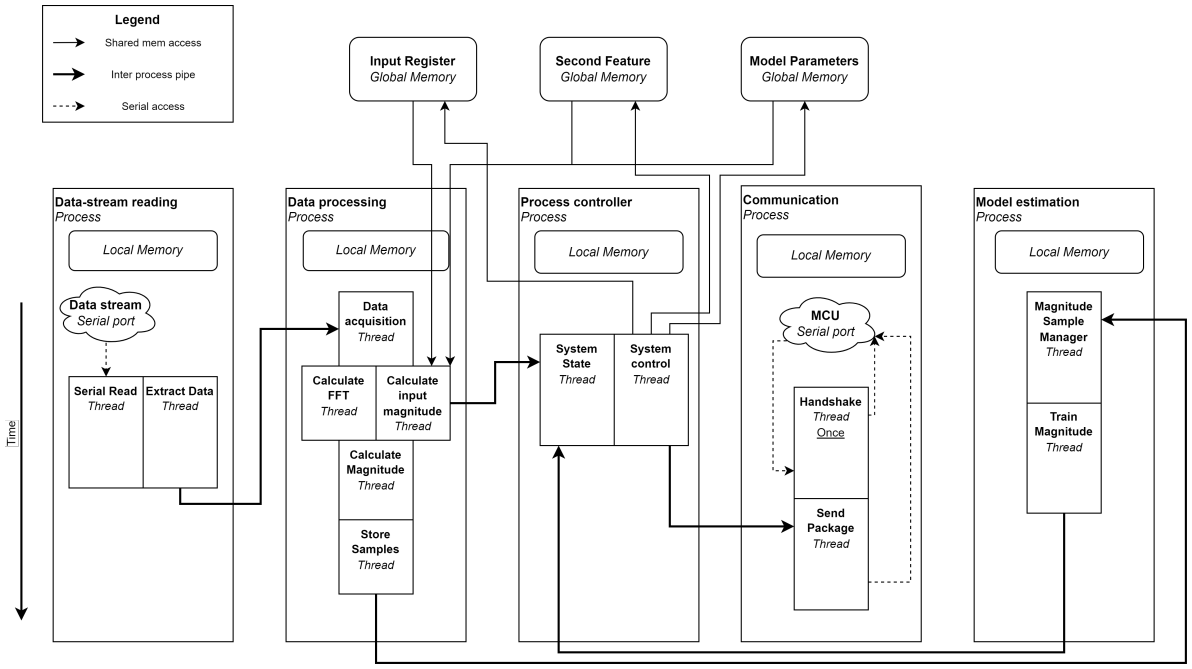



Figure 7.2: Data processing and Model estimation as multiprocessing module

**Main**

Besides the processes that are implemented, a main script is required for setting up the processes and running a GUI. The user interface is implemented as a GUI which is in fact a multi threaded implementation on another process. However, this is built in by the libraries that are used so this will not be treated as a separate process that is implemented.

The GUI is requesting the frequencies of the user input and the corresponding amplitudes. In this way, the user is able to define the wave-form that is the input of the ultrasound system. Besides the wave-form input, the GUI is also providing the option to perform a frequency sweep. Another feauture of the GUI is that it shows the magnitude samples in a graph and updates it when the samples are updated. When the model estimation is updated, it will be also made visible in the graph.

**Shared Memory**

In order to be able to share memory, shared memory needs to be created and referred to the processes that need access to the memory. However, the amount of shared memory should be kept as low as possible, since acquiring shared memory is in fact blocking other processes when they need to acquire it at the same time. This influences the run-time and thus quality of the whole subsystem. Shared memory is implemented as virtual memory on the PC. A virtual memory address will be created that is referenced to a physical address on the PC. The processes will be using this virtual address to access the data.

Three shared memory addresses are implemented namely memory containing the model parameter history, memory containing the input history and memory containing the second feature parameter history. The history of those parameters are required for calculating the input power that was on the input of the ultrasound system at a certain time. This time is stored as ID. When measurements are transmitted to the PC from the FPGA, the data is coupled to an ID, so that the input data can be generated on the PC and is not required to sent over the data-stream.

**Data-stream reading Process**

The primary task of the data-stream reading process is to read the data-stream that is received via a com-port. This process is implemented as two threads. The first thread is reading the com-port continuously and storing the received data. The data that is received consist of a sequence of packages

of two bytes. Each package (two bytes) represents a data type. There are three different data types. Namely the model ID, the second model feature and the position data. These packages are transmitted by the FPGA in sequences that will contain as much position data-points as the window-size is specified. The sequence always starts with one ID-package followed by one second model feature package and window-size amount of position-packages.

Consequently, the second thread will be extracting the actual data from the received packages. The packages are 2 bytes from which the first two bits are used to identify the package type. The remaining 14 bits contains the actual data content. So, this thread will identify the package type, convert the data content to decimal and transmit the data to the data processing process via a Pipe. There is also a feature implemented that recognizes data that is not conform the protocol. Also, this thread is only extracting data if it is demanded by the data-processing process. This leaves the serial read thread uninterrupted as possible.

**Data-processing Process**
The data-processing is implemented as a process of five threads. The fist thread is acquiring the data from the data-stream reading process. If the data is successfully received it will trigger two other threads to start. These two threads are the FFT calculation thread and the input estimation thread. After windowing the system output data that is received via the data stream, a FFT will be performed. The other thread is calculating the magnitude that corresponds to this output data by looking up the model state using the model ID. When both threads are finished, the magnitude response for each frequency component will be calculated. This results in a magnitude response sample and this will be transmitted to the model estimation process. That process stores the magnitude response samples and consequently estimated the model parameters.

**Process-controller Process**
The process-controller is performing control-like tasks. This process is implemented as two threads, namely the system state thread and the system control thread. The system state refers to the ultrasound system state and is keeping track of the current state by checking what the current ID is. When it receives a current ID, it will remove all older ID's from the input register since those ultrasound system states are already in history, so they will never be received again. Thus the input state of that specific moment in time is redundant. Besides keeping track of the current ID, it is generating new ID's and assigning the newly updated model parameters to an ID. It is generating these new ID's by simply generating random numbers between $0$ and $2^{14}$ and check whether the ID's are already in use by checking the input register. This is to prevent duplicate ID's.

The system control thread is according to the current state, so depending on the current ID, collecting all parameters needed by the FPGA to drive the new model and new frequencies when it is demanded. So, when the PC is triggered to send new frequencies or just an updated model, this thread is collecting all required parameters and transmitting it to the communication process that will transmitting it to the microcontroller.

**Communication Process**
The communication processes is responsible for transmitting the model parameters and user input to the microcontroller which is forwarding the data to the FPGA. The communication between microcontroller and pc is again over serial. The process is implemented as two threads. The first thread is only performed once at start-up. It will send a start-up message to the microcontroller and waits for the microcontroller to react. This handshake is implemented because otherwise it could occur that the microcontroller is still booting. After the handshake is completed, the system control process is triggered to proceed.

The second thread is compiling the messages transmitting via a com-port. The protocol is designed to distinguish between message types. A message always starts with a message-byte that specifies the type of the message. Consequently, depending on the message type, the content will be transmitted. The content length is specified by the message type and therefore the FPGA is expecting a specific number of bytes depending on the message type. A sequence of messages is always ended with the update-model message.

**Model estimation Process**
The model-fitting process consists of two threads. The first thread is the sample manager. Due to constant evaluation of the current state of the ultrasound system, the magnitude samples are constantly updated resulting in two problems. The first problem is that the memory of the hardware is limited. The second problem is that when it is decided to store all samples, the influence of the history of the ultrasound state will be dominant compared to the new state when the system is running for a relative long period of time. So, the sample manager is updating the magnitude samples by creating a memory that is divided into frequency and second-feature bins. Each bin is has a maximum bin-length and is implemented as a FIFO buffer. Each new sample is stored in the corresponding bin and if this bin exceeds the maximum bin length, the oldest sample will be deleted from the bin. After a specified amount of new samples are added to the sample memory, it will trigger the thread that is estimating the model parameters.

The second thread is performing the model estimation by optimizing for the least squares. The resulting model parameters represent the current state of the ultrasound system and thus need to be transmitted to the FPGA. So, after new model parameters are obtained, the thread will transmit them via a Pipe that is connected to the system state thread located in the process-controller process. That process will be triggered by new data in the Pipe and react accordingly.

## 7.3. Microcontroller

On the microcontroller multiple functions are implemented. The first function is a bridge for the UART communication between the PC and the FPGA. The second function is measuring the DC output response and calculating the optimal setpoint together with the PC. The last function is a control system to stabalize the setpoint at the right place it has calculated before with the pc. Due to the fact that at most two processes are running at the same time in the microcontroller, a dual core microcontroller is chosen. The microcontroller should also contain an analog to digital converter and a digital to analog converter for measuring voltage levels. It should also be able to communicate with a PC. In the end the ESP32 devkit board 1 [**ESP32**] has been chosen as a microcontroller. It has all the functions that are described above. Later the DAC from the microcontroller is exchanged for an external DAC, to improve the resolution of the analog voltage output.

**Calibration**
To calibrate the analog to digital converter(from now on: ADC) and the digital to analog converter(from now on: DAC) first measurements have to be done to see how much the output differs from the set voltage in bits. This is done by applying a voltage on the DAC output and measuring it with a multi meter for many different levels in the range of use. With this data, you can characterize the difference in set voltage and the real output. This is done by a least square fit to minimize errors. This will result in a function to be able to convert a voltage to bits so the DAC will output the correct voltage, or for the ADC: that it will read the correct voltage. To verify this, the input and output voltage have to be compared with a device that is able to measure voltage levels precisely, for instance a multi meter, for the DAC or a precise voltage source for the ADC. For instance a multi meter. This resulted in:

| DAC | ADC |
|---|---|
| No correction | offset: + 127 bits |

Table 7.1: Calibration results

### 7.3.1. Wavelength of the laser-diode as operating point

As explained in section 4.2 the wavelength of the laser diode must be controlled to achieve a phase error less then 0.1 radians. By sweeping the voltage at the set point input the waveform of the interference pattern is expected to be outputted at the DC output. Because in this way the wavelength is adjusted and thus the intensity of light at the detector following Equation 4.6. A system should be made to first detect the quadrature points of this wave and then controlling it to avoid drifting of the point.

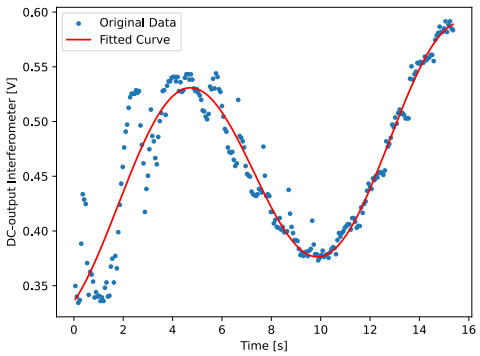
### 7.3.2. Quadrature point estimation: implementation and verification

Several methods for estimating the quadrature point are suggested in subsection 6.3.2. From the four suggestions, two methods are selected based on the criteria. The extreme value method is selected because it can be implemented isolated from the system on one of the two cores of the esp32. However, for this method it was assumed that the signal would be a sine-wave without an offset and constant amplitude. By measuring the interference pattern while sweeping the input voltage, it seemed that the assumption was not correct.

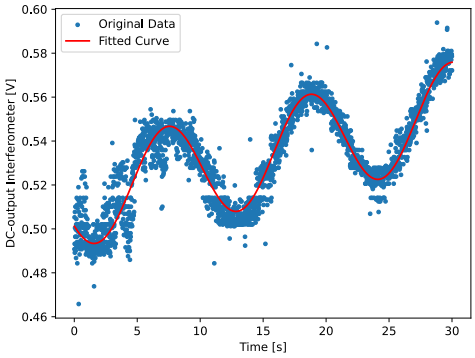
The second method selected is by performing the same measurement resulting in a sine-wave and consequently fitting a sine function in it. This method is implemented successfully by optimizing the following function to the measurements:

$$A * sin(2\pi f x + \phi) + B + Cx \tag{7.1}$$

Where x represents the x axis and all other variables are optimized. Optimization is implemented by least square fitting, so the least square as cost function is minimized. Before optimizing, initial values should be determined carefully otherwise the cost function will optimize for a local minimum that will converge to the wrong parameters. The most important parameter to initialize correctly is the frequency. If that is done correctly, the sine-fit will be successful. Results are shown in Figure 7.3. In Figure 7.4 the second derivative is illustrated of the optimized function given in Equation 7.1, with the zero-crossings being the quadrature points. Note that for all measurements it is related to time. However, to make this method useful, the x-axis should represent the control signal in voltage, to derive the correct operating point. And another downside for this implementation is that the sine-fitting is implemented on the PC which means that the program-flow of the overall system needs to be interrupted.




(a) Sine fit result using internal DAC with as input a ramp voltage from 0.6V till 0.8V




(b) Sine fit result using external DAC

Figure 7.3: Interferometer DC-output measurement data as result of a ramp input from 0.7V till 0.8V, fitted on a sine

In order to choose the optimal quadrature point, it is important to select the quadrature point that is closest to the temperature of the environment it is in. This is an assumption made based on measurement results after the first test. The first test results showed big fluctuations at a temperature of 310 kelvin till 303 kelvin. The assumption was made that the temperature in the room where the measurements were taken was around 293 kelvin (room temperature). In the plot it can be seen that there are less fluctuations at higher voltages and thus lower temperatures. These temperatures were near 293 kelvin. The voltage is plotted against the temperature in Figure 7.5. The voltage that was put in and the derivative sign are also saved. This information will later be used by the stabilizing system.

### 7.3.3. Stable operating point

To make sure the the operating point stays at the same level, it was decided to make a PID controller. The first implementation was a PI controller implemented on the ESP32 devkit microcontroller. This implementation was a discrete controller. The controller has multiple functions to protect the laser module from harmful temperatures such as a minimum voltage output and a maximum voltage output. This limits it's possibilities to obtain a stable response, but still a stable response can be obtained. Furthermore the input voltage of the set point is used as starting control signal to the laser module.

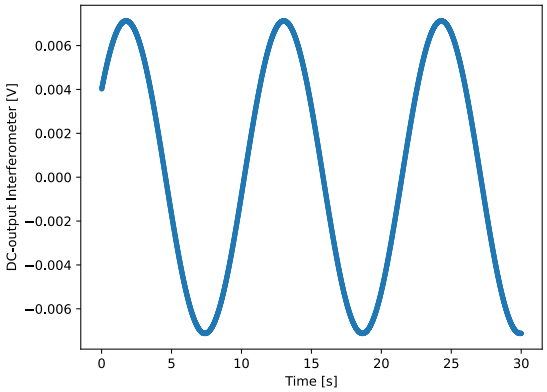



Figure 7.4: Second derivative with zero-crossings being quadrature points

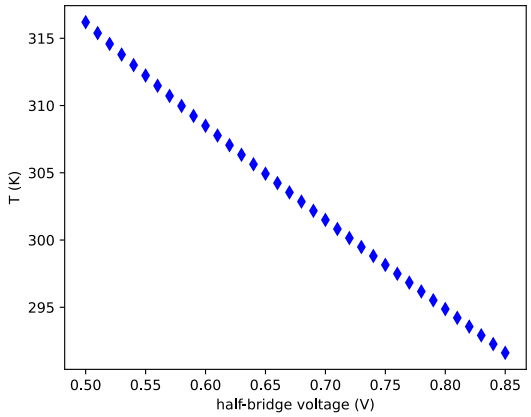



Figure 7.5: Voltage over temperature, measured by Warner form Quantified Sensor Technology

This is done to make sure that the operating point does not stabilize to another point in the wave with the same voltage level. Also the derivative sign is used, to make sure that the PID controller does stabilize the system instead of destabilizing. To obtain a stable response as fast as possible and as stable as possible, a simulation of the system should be made with a system model. With this system model and the system parameters: delay of the response and cut off frequency of the feedback signal, the parameters for a PID controller can be calculated. The delay can be obtained from a step response as shown in Figure 7.7.For now, due to time issues the controller is tuned by hand. After tuning the response was as given in Figure 7.6.

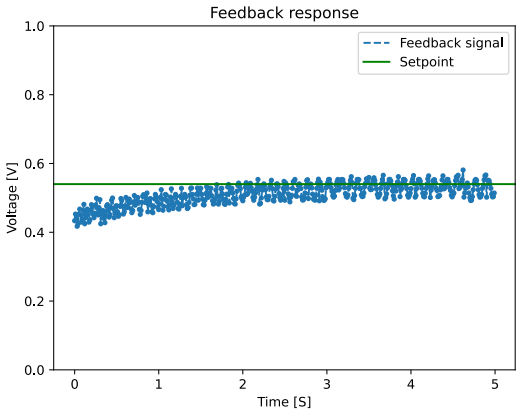



Figure 7.6: PID response for handmade tuning

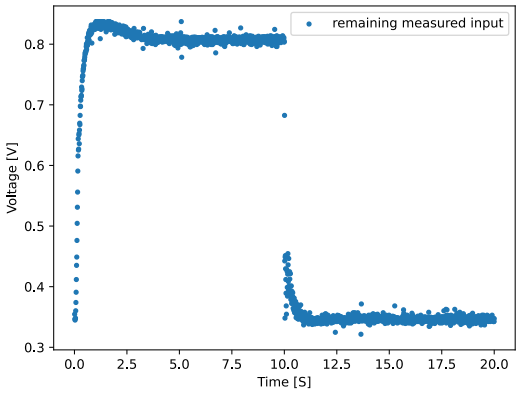



Figure 7.7: Step response

# 8

# Discussion and Conclusion

After investigating possible solutions for dynamically estimating the model of the ultrasound system using position measurements, a robust data-processing and model optimization technique is implemented. The power extraction in the frequency domain resulted in a theoretical improvement of the signal to noise ratio. However, because this method is in the frequency domain it is by definition not able to detect non-linear behaviour. So, hysteresis can not be detected by this method. By means of implementing a second feature to the model, a model is achieved which is in fact capturing non-linear behaviour if the second feature is chosen well.

According to the program of requirements, the steady-state ripple contribution of this subsystem can be at max 0.5dB. This requirement is not validated yet. However, assuming the design objectives of the ultrasound system are achieved, the model optimization should be performing well enough to fit the shape of the magnitude response.

Regarding the other main task of this subsystem, namely controlling the operating point of the interferometer, four solutions for estimating the quadrature-point are proposed and two are implemented. Wrong assumptions were made for the extreme value detection method, that became clear after performing measurements. The second method is validated to fit the properly and the quadrature point can be derived from the second derivative of that function. This method appeared to work out better than expected since this method gives the possibility to select a quadrature point closest to the environment-temperature.

In addition, this subsystem is integrated and tested as component. The system seems to be functioning as expected. However, a full system integration is not performed yet, which is likely to raise still undefined problems.

## 8.1. Future work

Not all necessary validations are performed to conclude that the requirements are met. However, promising results are achieved thus far. To succeed this project, parts of subsystem validation must be performed and full system integration has to be performed and validated. The work that has to be done is listed below:

- Validate data-processing and model estimation to meet the program of requirements by obtaining the maximum gain error and the SNR improvement.
- Full system integration.
- Obtain system model with its parameters and calculate optimal PID controller parameters

# Bibliography


[1] Don Berlincourt. “Piezoelectric Crystals and Ceramics”. In: *Ultrasonic Transducer Materials*. Ed. by O. E. Mattiat. Boston, MA: Springer US, 1971, pp. 63–124. ISBN: 978-1-4757-0468-6. DOI: 10.1007/978-1-4757-0468-6_2. URL: https://doi.org/10.1007/978-1-4757-0468-6_2.

[2] Anton V. Nikolaev et al. “Quantitative Evaluation of an Automated Cone-Based Breast Ultrasound Scanner for MRI–3D US Image Fusion”. In: *IEEE Transactions on Medical Imaging* 40.4 (2021), pp. 1229–1239. DOI: 10.1109/TMI.2021.3050525.

[3] Amirhossein Shahshahani, Sharmistha Bhadra, and Zeljko Zilic. “A Continuous Respiratory Monitoring System Using Ultrasound Piezo Transducer”. In: *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2018, pp. 1–4. DOI: 10.1109/ISCAS.2018.8351359.

[4] Mijia Yang and Pizhong Qiao. “Modeling and experimental detection of damage in various materials using the pulse-echo method and piezoelectric sensors/actuators”. In: *Smart Materials and Structures* 14.6 (Sept. 2005), p. 1083. DOI: 10.1088/0964-1726/14/6/001. URL: https://dx.doi.org/10.1088/0964-1726/14/6/001.

[5] Joo Young Pyun, Young Hun Kim, and Kwan Kyu Park. “Design of Piezoelectric Acoustic Transducers for Underwater Applications”. In: *Sensors* 23.4 (2023). ISSN: 1424-8220. DOI: 10.3390/s23041821. URL: https://www.mdpi.com/1424-8220/23/4/1821.

[6] Worapol Tangsopha, Jatuporn Thongsri, and Wutthikrai Busayaporn. “Simulation of ultrasonic cleaning and ways to improve the efficiency”. In: *2017 International Electrical Engineering Congress (iEECON)*. 2017, pp. 1–4. DOI: 10.1109/IEECON.2017.8075747.

[7] Xinxin Liu et al. “Off-Resonance Dynamics Behavior of Piezoelectric Micromachined Ultrasonic Transducer”. In: *2019 20th International Conference on Solid-State Sensors, Actuators and Microsystems & Eurosensors XXXIII (TRANSDUCERS & EUROSENSORS XXXIII)*. 2019, pp. 873–876. DOI: 10.1109/TRANSDUCERS.2019.8808621.

[8] K. A. Ahmad et al. “Design and Characterization Piezoelectric Acoustic Transducer for Sonar Application”. In: *2018 8th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*. 2018, pp. 233–237. DOI: 10.1109/ICCSCE.2018.8685014.

[9] Ashwin Sampathkumar and Jeffrey A. Ketterling. “Optical characterization of surface-displacement fields of high-frequency transducers”. In: *2014 IEEE International Ultrasonics Symposium*. 2014, pp. 1968–1970. DOI: 10.1109/ULTSYM.2014.0489.

[10] S. Sherrit et al. “Characterization of transducers and resonators under high drive levels”. In: *2001 IEEE Ultrasonics Symposium. Proceedings. An International Symposium (Cat. No.01CH37263)*. Vol. 2. 2001, 1097–1100 vol.2. DOI: 10.1109/ULTSYM.2001.991910.

[11] N. Barker and T. Evers. “An Integrated System for Live Characterization and Adaptive Compensation of Ultrasonic Transducers”. Bachelor thesis. Delft university of technology, June 2023.

[12] M. Vermeulen and N. van Klaveren. “Amplifier design for a piezoelectric transducer”. Bachelor thesis. Delft university of technology, June 2023.

[13] Xingchun Chu, Haibao Lü, and Shanghong Zhao. “Research on long-range grating interferometry with nanometer resolution”. In: *Measurement Science and Technology* 19.1 (2007), p. 017001.

[14] M Yu Vyatkin, Semen Pavlovich Grabarnik, and Oleg A Ryabushkin. “Temperature dependence of the radiation wavelength of a fibre laser”. In: *Quantum Electronics* 35.4 (2005), p. 323.

[15] Julie Carreaud et al. “Wavelength switching in Nd: YSAG ceramic laser induced by thermal effect”. In: *Laser Physics Letters* 9.5 (2012), p. 344.

[16] *Software Filtering: Windowing -General Analog Concepts*. 2022. URL: https://www.ni.com/nl-nl/support/documentation/supplemental/06/software-filtering--windowing---general-analog-concepts.html.

[17] Peter Schaldenbrand. *Window Correction Factors*. 2019. URL: https://community.sw.siemens.com/s/article/window-correction-factors.

[18] Xuefeng Mao et al. “Stabilized Fiber-Optic Fabry–Perot Acoustic Sensor Based on Improved Wavelength Tuning Technique”. In: *Journal of Lightwave Technology* 35.11 (2017), pp. 2311–2314. DOI: 10.1109/JLT.2017.2651151.

[19] Qiaoyun Wang and Zhenhe Ma. “Feedback-stabilized interrogation technique for optical Fabry–Perot acoustic sensor using a tunable fiber laser”. In: *Optics Laser Technology* 51 (2013), pp. 43–46. ISSN: 0030-3992. DOI: https://doi.org/10.1016/j.optlastec.2013.03.019. URL: https://www.sciencedirect.com/science/article/pii/S0030399213001023.

[20] Xuefeng Mao, Xinlei Zhou, and Qingxu Yu. “Stabilizing operation point technique based on the tunable distributed feedback laser for interferometric sensors”. In: *Optics Communications* 361 (2016), pp. 17–20. ISSN: 0030-4018. DOI: https://doi.org/10.1016/j.optcom.2015.10.022. URL: https://www.sciencedirect.com/science/article/pii/S0030401815302066.

[21] Xunqiang Gong et al. “Comparative Analysis of three outlier detection methods in univariate data sets”. In: *2022 3rd International Conference on Electronic Communication and Artificial Intelligence (IWECAI)*. 2022, pp. 209–213. DOI: 10.1109/IWECAI55315.2022.00048.