



ELEKTRO- UND INFORMATIONSTECHNIK
MECHATRONIK
REGENERATIVE ENERGIETECHNIK UND ENERGIEEFFIZIENZ

Deckblatt

Daten des Praktikanten

Name: Marius Graml
geboren am: 18.09.1999
Email: marius.graml@st.oth-regensburg.de
Semesterbezeichnung: EI7b
Matrikelnummer: 3233111

Praktikantenamt der
OTH Regensburg

Hr. Krause
Prüfeningerstr. 58
93049 Regensburg
Tel.: 0941/943-1043

Firmendaten

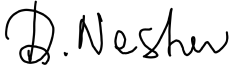
Name, Anschrift der Firma: Australian National University, Canberra
Abteilung(en): TMOS
Praktikumsbetreuer: Dr. Dragomir Neshev / Dr. Roland Schiek
Email: dragomir.neshev@anu.edu.au
Tel.: (02) 612 53792

Praktikumsbericht EI ☒ ME ☐ RE ☐

Dauer des Praktikums: von 19.09.2022 bis 24.02.2023

Anzahl der Wochen: 23 Fehltage: 0

Bestätigung des Praktikumsbetreuers, dass er den Bericht gelesen hat und mit dem Inhalt einverstanden ist.


(Unterschrift Betreuer bei der Firma)

Australian National University
(Stempel der Firma)


(Unterschrift Praktikantin/Praktikant)

List of Activities

Date / Duration		Description
19/09/2022 – 21/09/2022	3 days	Lab inductions and safety briefings
22/09/2022	One day	Receiving task for project 1: Implementing MATLAB function which determines the auto convolution of a spectrum measured by a FROG
23/09/2022 – 29/09/2022	7 days	Implementing MATLAB function
30/09/2022	One day	Receiving task for project 2: Setting up a Michelson interferometer that can adjust itself automatically to a specific fringe pattern. Therefore, a Moku:Go device has to be used respectively its embedded PID controller.
01/10/2022 – 24/10/2022	24 days	Gathering information/literature for the project. This means: <ul style="list-style-type: none"> - Understanding the functionality of a Michelson interferometer - Understanding the behavior of the light within the interferometer (constructive and destructive interference, measured intensity on monitor) - Understanding the functionality of the Moku:Go and its PID controller - Figuring out how the PID controller can be addressed using Python - Understanding the whole control system, which has to be set up, and its difficulties

25/10/2022 – 01/11/2022	8 days	<ul style="list-style-type: none"> - Setting up the Michelson Interferometer manually in the lab
02/11/2022 – 02/12/2022	31 days	<ul style="list-style-type: none"> - Implementing a user interface using Python to access both Moku:Go's provide parameter values for each run of the system
03/12/2022 – 10/12/2022	8 days	<ul style="list-style-type: none"> - Testing the setup in combination with the user interface
11/12/2022 -19/12/2022	9 days	<ul style="list-style-type: none"> - Fixing / solving issues of the setup and the user interface
20/12/2022 – 23/01/2023	35 days	<ul style="list-style-type: none"> - Trying to solve general issues of the project
24/01/2023 – 08/02/2023	16 days	<ul style="list-style-type: none"> - Checking out the app which is provided by the Moku:Go and trying to set up the control system by using the app
09/02/2023 – 17/02/2022	9 days	<ul style="list-style-type: none"> - Comparing results of the own software with the results of the app
18/02/2023 – 24/02/2023	7 days	<ul style="list-style-type: none"> - Finalising report - Removing equipment from optical table
	Σ 23 weeks	

Internship Report

Marius Graml

OTH Regensburg
The Australian National University

Contents

1	The Project	3
1.1	The Functionality of a Michelson Interferometer	3
1.2	The Difficulty of Controlling the Interferometer	6
1.3	Obtaining the First Derivative of the Intensity	9
1.4	Setting Up the Controlling System	10
	List of Equipment	10
	Controlling Loop	10
1.5	The User Interface	12
	Main Screen	12
	Oscilloscope Menu	12
	Dither Menu	13
	Filter Menu	14
	PID Controller Menu	17
	Controlling Menu	18
2	Problems and Solutions	20
	Problem 1	20
	Problem 2	20
	Problem 3	21
	Problem 4	21
	Problem 5	21
3	Results	22
3.1	Controlling by using the App	22
3.2	Controlling by using the Software/Python API	24

This report is about my internship at the ARC Centre of Excellence for Transformative Meta-Optical Systems (TMOS) at the Australian National University in Canberra.

1 The Project

The aim was to set up a Michelson Interferometer that is able to adjust itself automatically to a specific fringe pattern. More precisely, the interferometer is supposed to remain in a state in which destructive interference can be seen on the screen. To understand the idea behind the setup of the project and its difficulties, a closer look at the functionality of a Michelson Interferometer itself and the physics behind it has to be taken.

1.1 The Functionality of a Michelson Interferometer

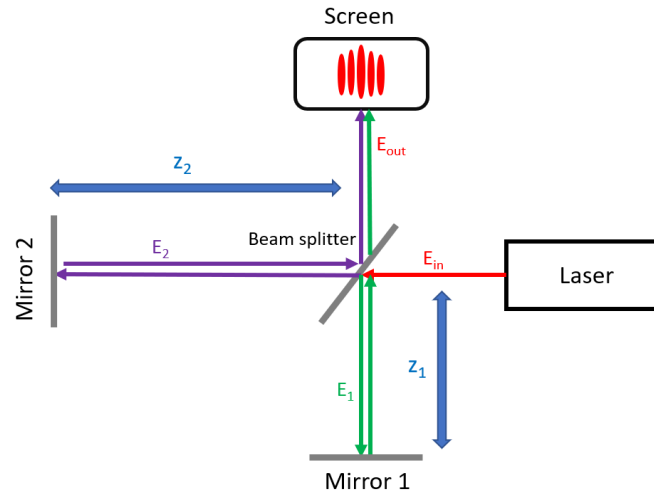


Figure 1: Functionality Michelson Interferometer

A laser beam (E_{in}) is sent to a beam splitter which divides the beam 50:50 into two equivalent beams with the same intensity. The one beam goes through the beam splitter first, is reflected by Mirror 2, then reflected by the beam splitter, and finally sent to a screen. The second beam goes through the same process but in the opposite order. Both beams interfere on the screen which leads to a specific fringe pattern. The form

of the pattern depends on the angle of the mirrors. In case the mirrors are exactly perpendicular to each other, a circular fringe pattern, like this in Figure 1, can be seen. Another condition, which influences the pattern, is the distance of the mirrors to the beam splitter. Depending on the distance, the beams need different times to reach the screen and therefore have different absolute phases when they reach it. In other words, depending on the path difference of mirrors 1 and 2 the beam waves have a specific phase difference when they reach the screen. Depending on the phase difference, the waves interfere constructive or destructive and lead to different interference patterns on the screen.

Mathematically it can be described as follows.

Two waves with the same direction of propagation are considered whereby the waves have different phases (Eq.1).

$$\begin{aligned} E_1(t) &= E_1 * \cos(\omega t - kz + \phi_1) = E_1 * \cos(\theta_1(t)) \\ E_2(t) &= E_2 * \cos(\omega t - kz + \phi_2) = E_2 * \cos(\theta_2(t)) \end{aligned} \quad (1)$$

Since both waves have the same frequency, ωt can be seen as constant. Therefore, the absolute phase difference with respect to the path difference can be determined as follows (Eq. 2):

$$\Delta\theta(\Delta z) = \theta_1(t) - \theta_2(t) = \omega t - kz + \phi_1 - \omega t + k(z + \Delta z) + \phi_2 = \frac{2\pi}{\lambda}\Delta z + \phi_1 - \phi_2 \quad (2)$$

a) Constructive Interference

Constructive interference occurs if the absolute phases of the beam waves are the same. This means mathematically that the phase difference between both waves is an integer multiple of 2π . So, for constructive interference the following condition is necessary in terms of the path difference (Eq. 3):

$$\begin{aligned} \Delta\theta(\Delta z) &= \frac{2\pi}{\lambda}\Delta z + \phi_1 - \phi_2 = 2\pi m \quad \forall m = 0, +1, -1, +2, -2, \dots \\ \Delta z &= \left(m + \frac{-\phi_1 + \phi_2}{2\pi}\right)\lambda \\ \text{for } \phi_1 &= \phi_2 : \\ \Delta z &= m\lambda \end{aligned} \quad (3)$$

b) Destructive Interference

Destructive interference occurs if the phase difference between both waves is an integer

multiple of π . So, for constructive interference the following condition is necessary in terms of the path difference (Eq. 4):

$$\begin{aligned}\Delta\theta(\Delta z) &= \frac{2\pi}{\lambda}\Delta z + \phi_1 - \phi_2 = (2m+1)\pi \quad \forall m = 0, +1, +2, \dots \\ \Delta z &= \left(2m+1 + \frac{-\phi_1 + \phi_2}{\pi}\right)\frac{\lambda}{2} \\ &\text{for } \phi_1 = \phi_2 : \\ \Delta z &= m\lambda + \frac{\lambda}{2}\end{aligned}\tag{4}$$

Furthermore, the intensity of the interfered light beams on the screen must be determined.

Generally, the intensity of a light wave is defined as follows (Eq. 5):

$$I_{out} = \underline{E}_{out} * \underline{E}_{out}^* \tag{5}$$

According to Figure 1, \underline{E}_{out} describes the electric field of the interfered light beams and is therefore defined as follows (Eq. 6):

$$\begin{aligned}\underline{E}_{out} &= \underline{E}_{out1} + \underline{E}_{out2} \\ &= E_{in} * r * t * e^{j(\omega t - kz_1 + \phi_1)} + E_{in} * r * t * e^{j(\omega t - kz_2 + \phi_2)}\end{aligned}\tag{6}$$

So, \underline{E}_{out1} and \underline{E}_{out2} describe the light beams which come to the screen and interfere. \underline{E}_{in} describes the electric field of the original laser beam and r respectively t stands for the reflection respectively transmission factor of the beam splitter.

Considering the path difference $z_1 = z_2 + \Delta z$, Eq. 7 is obtained.

$$\begin{aligned}E_{out} &= E_{out1} + E_{out2} \\ &= E_{in} * r * t * e^{j(\omega t - k(z_2 + \Delta z) + \phi_1)} + E_{in} * r * t * e^{j(\omega t - kz_2 + \phi_2)}\end{aligned}\tag{7}$$

Since both laser beams \underline{E}_{out1} and \underline{E}_{out2} originate from the same laser beam \underline{E}_{in} , both beams have the same phase which means that both phases can be considered as $\phi_1 = \phi_2 = \phi = 0$. This delivers Eq. 8:

$$\begin{aligned}\underline{E}_{out} &= \underline{E}_{out1} + \underline{E}_{out2} \\ &= E_{in} * r * t * (e^{j(\omega t - kz_2 - k\Delta z)} + e^{j(\omega t - kz_2)}) \\ &= E_{in} * r * t * e^{j(\omega t - kz_2)}(e^{-jk\Delta z} + 1)\end{aligned}\tag{8}$$

Therefore, the following applies for the intensity (Eq. 9):

$$\begin{aligned}
I_{out} &= \underline{E}_{out} * \underline{E}_{out}^* \\
&= |E_{out}|^2 \\
&= |E_{in} * r * t * e^{j(\omega t - kz_2)} (e^{-jk\Delta z} + 1)|^2 \\
&= E_{in}^2 * r^2 * t^2 * |e^{j(\omega t - kz_2)}|^2 * |e^{-jk\Delta z} + 1|^2 \\
&= E_{in}^2 * r^2 * t^2 * e^{j(\omega t - kz_2)} * e^{-j(\omega t - kz_2)} * |e^{-jk\Delta z} + 1|^2 \\
&= E_{in}^2 * r^2 * t^2 * |e^{-jk\Delta z} + 1|^2 \\
&= 2 * E_{in}^2 * r^2 * t^2 * (\cos(k\Delta z) + 1)
\end{aligned} \tag{9}$$

Since the beam splitter splits up the intensity of the original beam equally, the reflection factor and the transmission factor are equal: $r^2 = t^2$. Due to the relationship of both factors as $r^2 + t^2 = 1$, both factors with power to 2 lead to the same value, namely $\frac{1}{2}$ (Eq. 10):

$$r^2 = t^2 = \frac{1}{2} \tag{10}$$

Finally, the intensity on the screen can be described with the following function (Eq. 11):

$$I_{out} = 2 * E_{in}^2 * \frac{1}{4} * (\cos(k\Delta z) + 1) = \frac{1}{2} * E_{in}^2 * (\cos(k\Delta z) + 1) \tag{11}$$

Figure 2 shows a plot of the intensity of the interfered light beams on the screen with respect to the path difference of the corresponding mirrors.

The red points describe the path differences where destructive interference occurs. These are the desired states to which the interferometer is supposed to be adjusted automatically. Since the function is symmetric, in each controlling process only a section with width of λ is considered. The green dashed lines demonstrate an example section.

1.2 The Difficulty of Controlling the Interferometer

Usually, error signals, i.e. the difference of the set point and the measured signal, equal zero at the desired value and are anti-symmetric about this point. This means that each value of the error signal can be assigned an unambiguously control direction. For example, by controlling a car to a specific speed, the error signal is either a positive or negative value. If the value is positive, the car is too slow and has to increase its speed. If the value is negative, the car is too fast and has to decrease its speed.

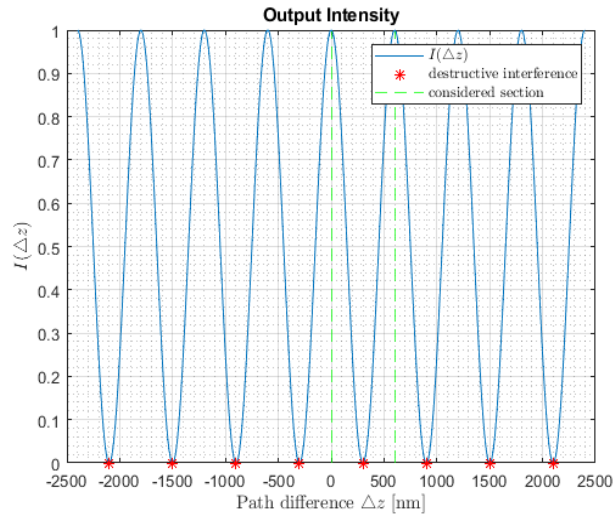


Figure 2: Output intensity on screen with respect to path difference

the plot of the controlling value (i.e. speed) over the variable (for example position of the accelerator in cm) could be as follows (Fig. 3):

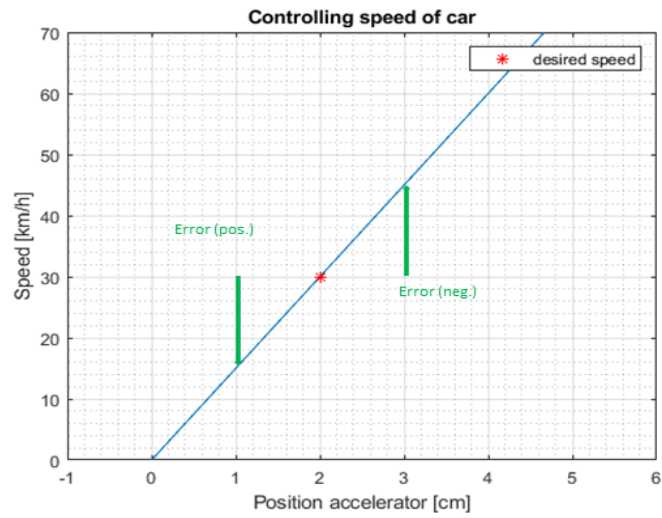


Figure 3: Example controlling car speed

So, each error value has an unambiguous corresponding control direction. However, that is not the case for controlling an interferometer to a specific interference pattern because in regards to the set point the function is symmetric and not anti-symmetric. Considering the relationship between the controlling value “intensity” and the variable “path difference” shows that for each error value exist two possible control directions in each considered section (Fig. 4). This means, that for each error value (except 0) it is unclear, in which direction the interferometer is supposed to be adjusted to reach destructive interference. This property leads to an unstable behavior of the whole controlling system.

The solution of that problem is using the first derivative of the intensity as control value. The reason why the first derivative is helpful is that in one considered section the first derivative is anti-symmetric (Fig. 5). This means that for each control direction exists only one error value. Consequently, for each error value it is clear in which direction to interferometer must be aligned (same as for the car example).

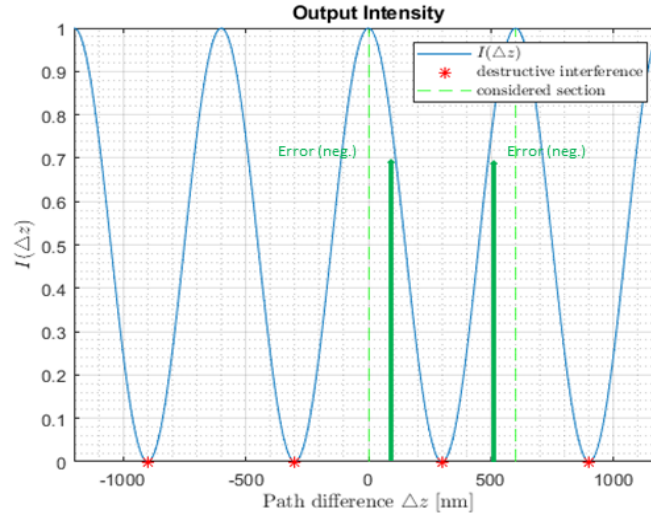


Figure 4: Controlling intensity on screen

For the first derivative of the intensity function follows (Eq. 12 and Fig. 5):

$$I'(\Delta z) = \left[\frac{1}{2} * I_{in} * \cos(k\Delta z) + 1 \right]' = -\frac{1}{2} * I_{in} * k * \sin(k\Delta z) \quad (12)$$

So, as it can be seen in Fig. 5, by using the first derivative of the intensity as control value, the desired value of the first derivative (i.e. the set point) is $I'(\Delta z) = 0$

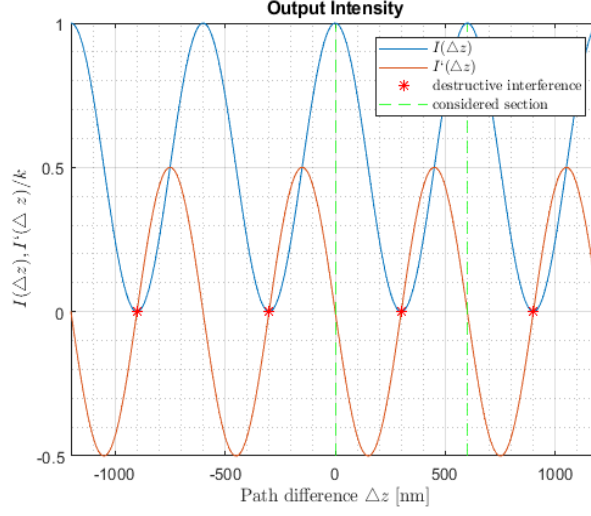


Figure 5: First derivative of intensity

1.3 Obtaining the First Derivative of the Intensity

A common technique to obtain the first derivative of a signal is applying a so-called dither signal which is in most cases a sine wave with amplitude A_{dith} and frequency ω_{dith} . This dither signal is added to the position of one of the mirrors so that the path difference oscillates around its DC-value Δz (Eq. 13).

$$\Delta z(t) = \Delta z_{DC} + A_{dith} * \sin(\omega_{dith}t) \quad (13)$$

Thus, for the intensity follows by describing the function with its Taylor series (Eq. 14):

$$I(\Delta z(t)) = I(\Delta z) + I'(\Delta z) * A_{dith} * \sin(\omega_{dith}t) + \frac{I''(\Delta z)}{2} * A_{dith}^2 * \sin^2(\omega_{dith}t) + \dots \quad (14)$$

Because of the very small amplitude of the dither signal it applies $A_{dith} \ll 1$. Therefore, in order to get a good estimation of the function it is sufficient to develop the Taylor series until the second element. After measuring the intensity on the screen by using a photodetector, the next step is to demodulate and finally filter the dither signal to extract the first derivative. For the demodulation, another sine wave is used which has the same frequency as the dither signal and a phase difference of θ . Considering the phase of the dither signal as zero, the demodulation signal is as follows (Eq. 15):

$$x_{demod}(t) = \sin(\omega_{dith}t + \theta) \quad (15)$$

Applying the demodulation by multiplying the demodulation signal to the measured intensity leads to the following expression (Eq. 16):

$$\begin{aligned} & [I(\Delta z) + I'(\Delta z) * A_{dith} * \sin(\omega_{dith}t)] * \sin(\omega_{dith}t + \theta) \\ &= I(\Delta z) * \sin(\omega_{dith}t + \theta) + I'(\Delta z) * A_{dith} * \sin(\omega_{dith}t) * \sin(\omega_{dith}t + \theta) \\ &= I(\Delta z) * \sin(\omega_{dith}t + \theta) + I'(\Delta z) * A_{dith} * \left[\frac{1}{2} \cos(\omega_{dith}t - \omega_{dith}t - \theta) - \frac{1}{2} \cos(\omega_{dith}t + \omega_{dith}t + \theta) \right] \\ &= I(\Delta z) * \sin(\omega_{dith}t + \theta) - \frac{1}{2} I'(\Delta z) * A_{dith} * \cos(2\omega_{dith}t + \theta) + \frac{1}{2} I'(\Delta z) * A_{dith} * \cos(\theta) \\ &= \frac{1}{2} I'(\Delta z) * A_{dith} * \cos(\theta) + \sigma(\omega) \end{aligned} \quad (16)$$

Here, $\sigma(\omega)$ describes all higher frequencies.

Finally, after removing the higher frequencies with a proper low pass filter, the following error signal is received (Eq. 17).

$$\begin{aligned} e &= \text{setpoint} - \text{measuredsignal} = 0 - \frac{1}{2} I'(\Delta z) * A_{dith} * \cos(\theta) \\ &= -\frac{1}{2} I'(\Delta z) * A_{dith} * \cos(\theta) \end{aligned} \quad (17)$$

1.4 Setting Up the Controlling System

List of Equipment

Table 1 shows the used equipment for this project.

Controlling Loop

The idea of the controlling system is depicted in (Fig. 6).

The intensity of the fringe pattern on the monitor is measured by a photodetector. Depending on the intensity, the photodetector generates an output voltage which is obtained as an input signal by one of the two Moku:Go's which is used as oscilloscope (Moku:Go 1). From this point the further steps will be processed digitally on a PC/laptop by accessing the Moku:Go's and the PZT-controller via Python. The measured input signal of the Moku:Go 1 is multiplied by a sine wave (demodulation signal) with a circular

Item	Amount
Moku:Go from Liquid Instruments https://www.liquidinstruments.com/products/hardware-platforms/mokugo/	2
Optical mirror with mirror adjustable mirror mount (KM100CP) from Thorlabs https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=1492	1
PZT-Mirror (KC1-T-PZ) from Thorlabs https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=231&pn=KC1T-P/M	1
Laser (CPS532-C2) from Thorlabs https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=1487	1
Aperture	1
BNC cables	2
USB-C cables	2
USB-hub (if necessary)	1
Beam splitter (50:50) from Thorlabs	1
DET10A\M – Si Biased Detector, 200 – 1100 nm, Thorlabs https://www.thorlabs.com/thorProduct.cfm?partNumber=DET10A/M	1
Piezo Controller (MDT693B) from Thorlabs with 3 outputs for all 3 axes https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_ID=1191	1
PC/Laptop with Python	1

Table 1: List of Equipment

frequency ω_{dith} and a phase θ . This is the frequency of the dither signal. θ is the phase difference of the dither signal and the demodulation signal in case the phase of the dither signal is considered as 0. After the demodulation, the signal can be described as $\frac{A_{dith}}{2} * \cos(\theta) * I'(\Delta L) + O(\omega)$. The signal contains the first derivative multiplied with a constant value but also several higher frequencies. These higher frequencies are removed by a digital low-pass filter in the next step. Since the setpoint of the controlling system is set to $I'(\Delta L) = 0$, the PID controller receives the negative output of the digital filter as input signal. The constant output signal of the PID controller is forwarded to the PZT-controller by using its USB connectivity (see manual). Additionally, the dither signal is added to the constant PID output by using the other Moku:Go as Waveform Generator and the extern BNC input of the PZT-controller (see manual). Finally, the PZT-controller amplifies its input voltage and forwards it to the PZT-mirror. This loop will be passed through repeatedly. All digital processes are done by one Python program.

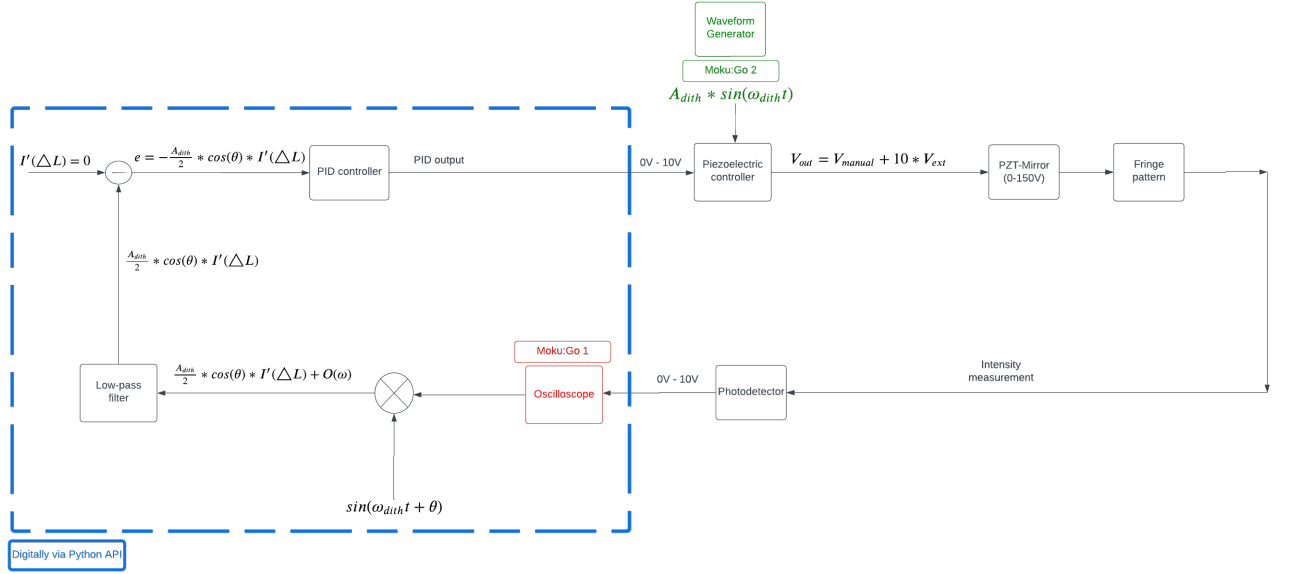


Figure 6: Controlling System

1.5 The User Interface

For accessing the Moku:Go's and the PZT-controller, inserting all necessary parameter values and observing the signals, a user interface was created using the PyQt5 package in Python. The user interface consists of one main screen and five sub-screens.

Main Screen

The main screen is the main menu of the interface (Fig. 7). From this menu every sub-screen or rather sub-menu can be reached just by clicking one of the buttons.

Oscilloscope Menu

In the oscilloscope menu (Fig. 8) the IP-address of the Moku:Go, which is used as oscilloscope, the input channel of the oscilloscope and the desired time base of the real-time plot must be provided. For choosing a time base, two values, separated by a space bar, must be inserted. The first value is not allowed to be less than zero and higher than the second value. By choosing the time base, the user can basically influence the x-axis of the signal plots of the controlling screen. By default, the input channel 1 and a time

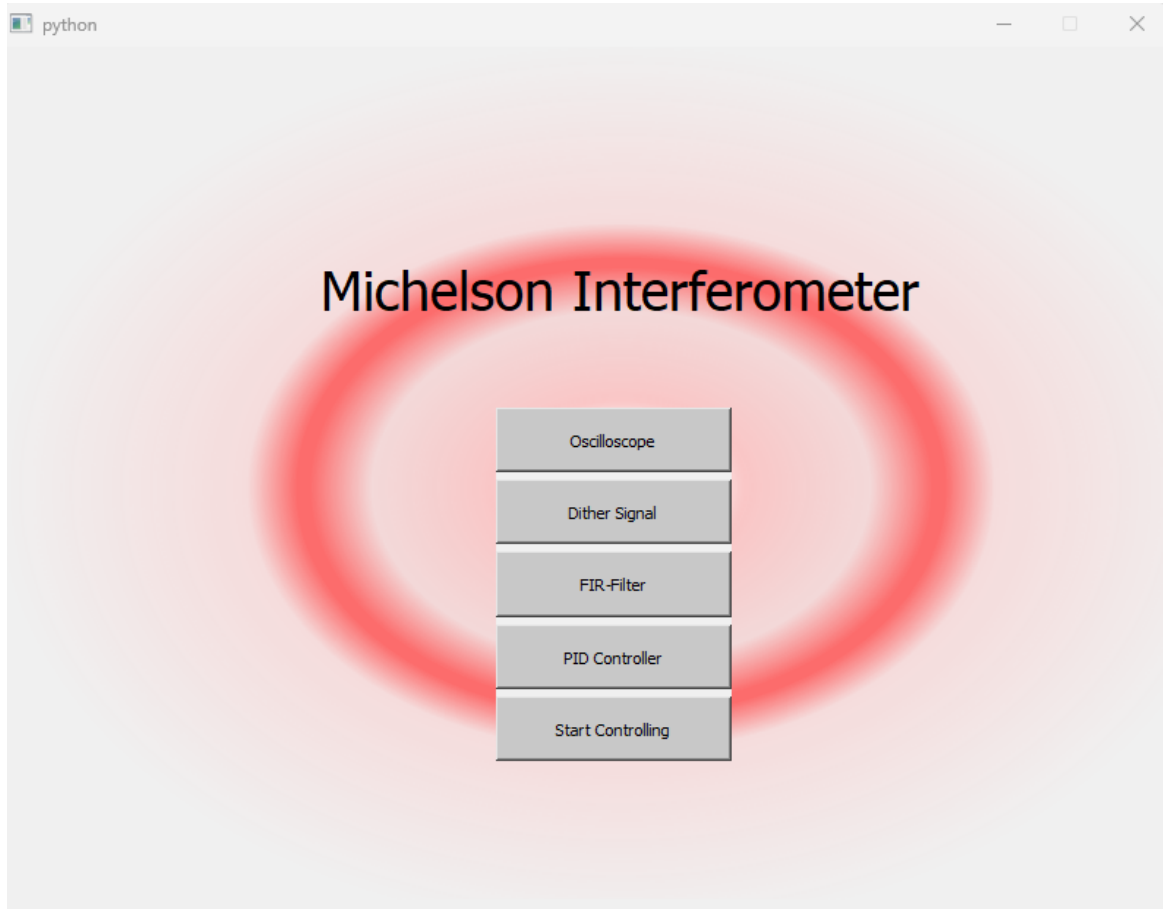


Figure 7: Main Screen

base of $[0, 8.192\text{e-}6]$ is chosen. After clicking the run button, the laptop connects to the Moku:Go and configures the oscilloscope.

Dither Menu

The dither menu (Fig. 9) is used to configure the other Moku:Go which is used as waveform generator. This means, the amplitude and frequency of the dither signal must be inserted. Additionally, the relative phase difference between the dither signal and the demodulation signal must be provided. Also, the user must choose the IP-address of

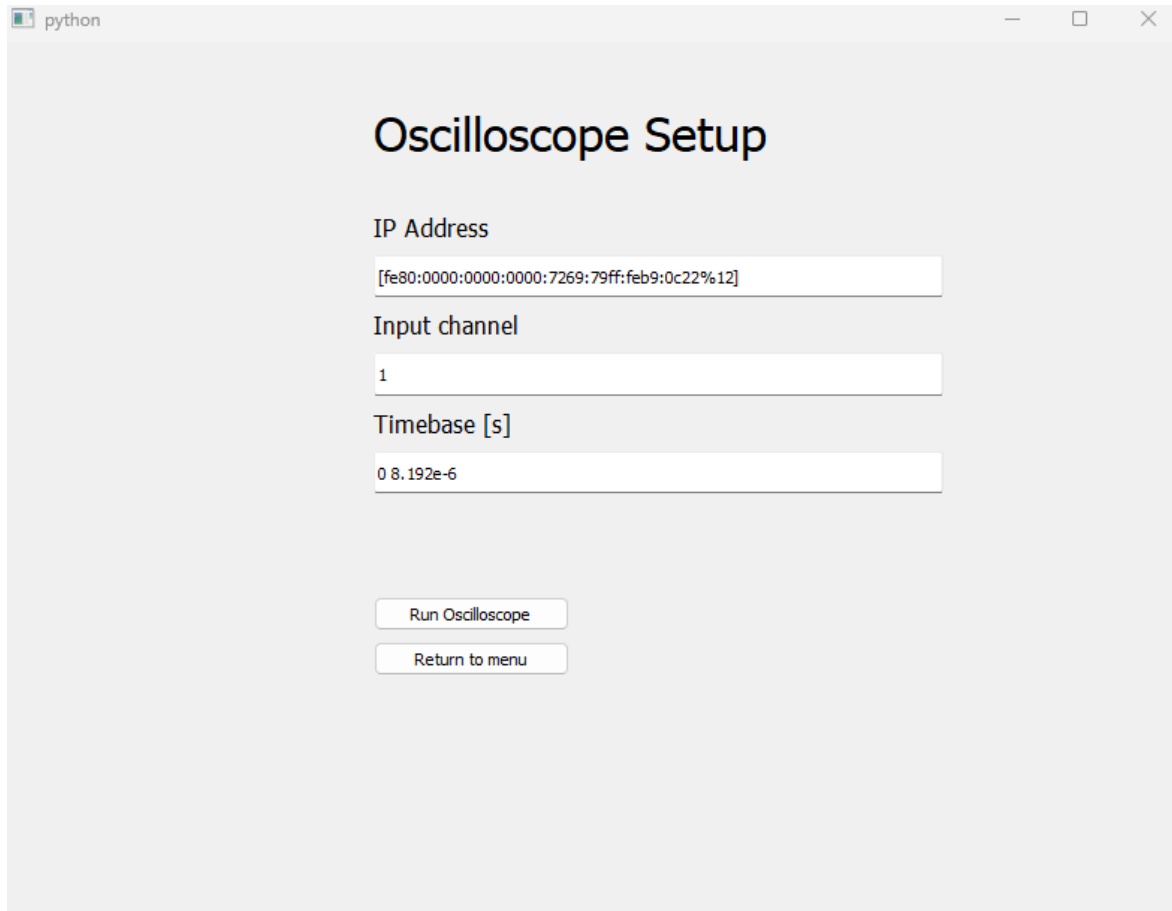


Figure 8: Oscilloscope Menu

the second Moku:Go and the output channel. Per default, channel 1 is used as output channel.

Filter Menu

The filter menu (Fig. 10) is used to choose the filter type for the low pass filter and the necessary filter parameters of the chosen filter type. The provided filter types are:

- FIR filter
- Butterworth filter

python

Waveform-Generator Setup

IP Address

[fe80:0000:0000:0000:7269:79ff:feb9:0b52%7]

Output channel

1

Frequency [Hz]

Phase difference [rad]

Amplitude [V]

Run Generator

Return to menu

Figure 9: Dither Menu

- Chebyshev 1 filter
- Elliptic filter

Depending on the chosen filter type, the following filter parameters must be provided:

- Filter order
- Cut-off frequency
- Max. ripple [dB]

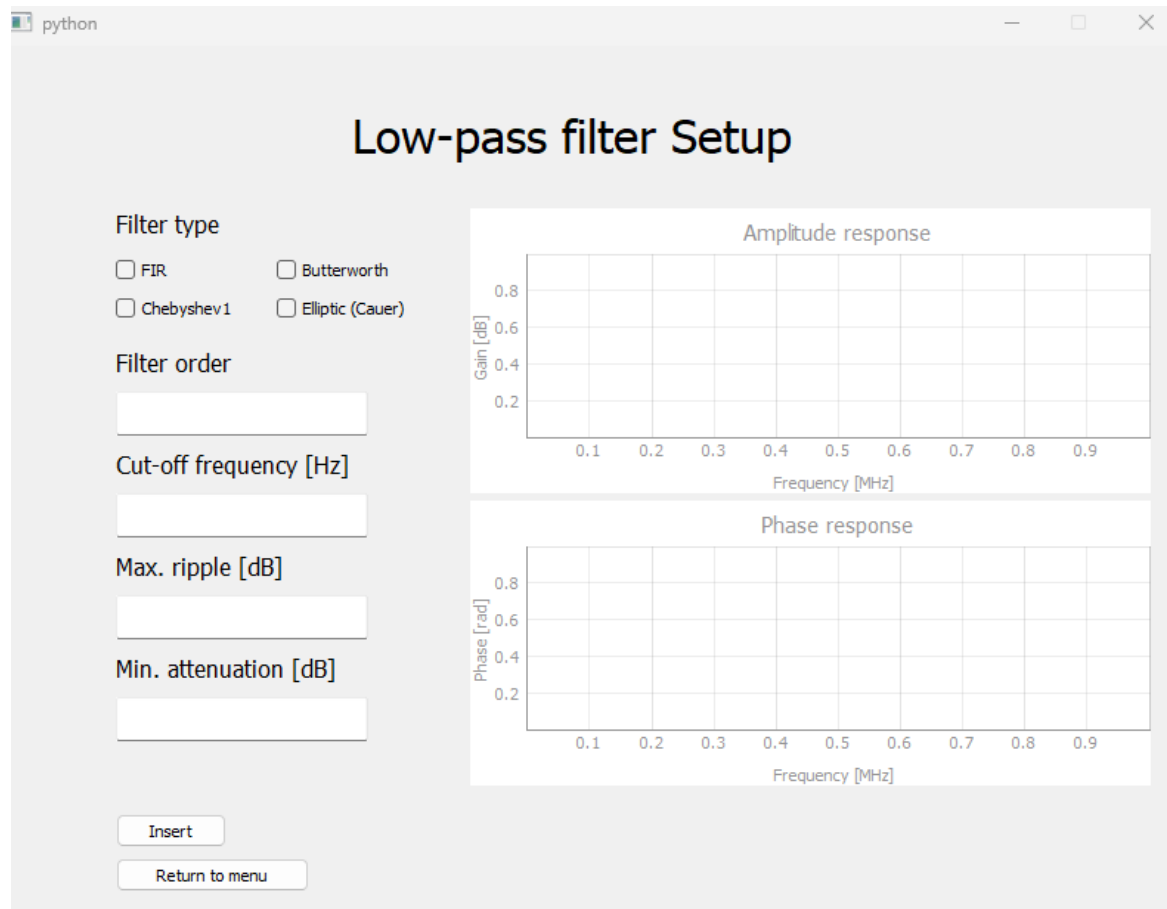
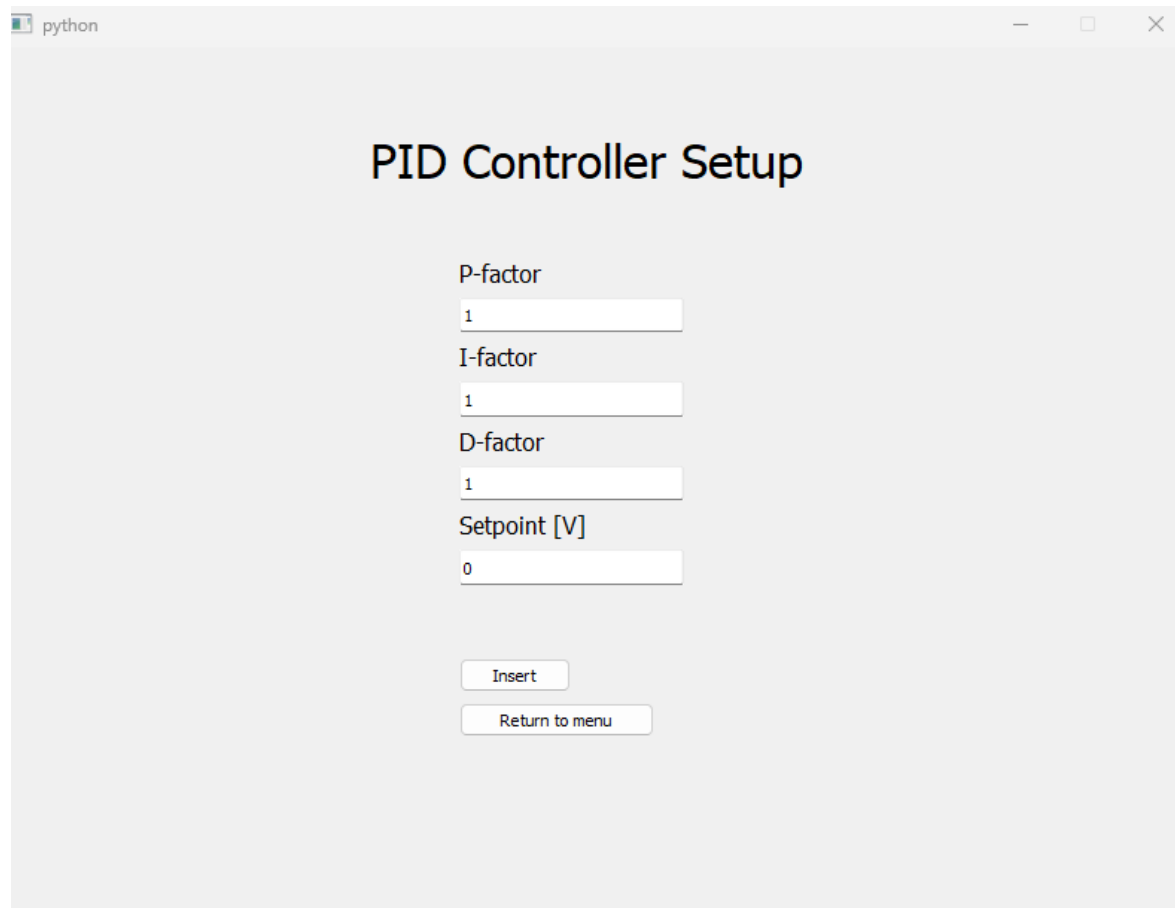


Figure 10: Filter Menu

- Min. attenuation

After clicking the insert button, the frequency response, i.e. amplitude and phase response, of the digital filter is depicted on the right side of the menu. It basically applies, the higher the filter order is chosen the better the filter fulfills the desired properties. However, this also leads to a longer computational time. Note that the oscilloscope must be running when the filter is chosen because the sample frequency of the oscilloscope must be determined first. Also, the maximum number of filter coefficients are 2000 for the FIR filter and 60 for all IIR filters. It is very likely that the IIR filters lead to an unstable behavior of the system.

PID Controller Menu



The image shows a screenshot of a Python application window titled "python". The window displays a "PID Controller Setup" menu. The menu consists of four input fields, each with a label above it: "P-factor", "I-factor", "D-factor", and "Setpoint [V]". Each input field contains the value "1". Below the input fields are two buttons: "Insert" and "Return to menu".

Parameter	Value
P-factor	1
I-factor	1
D-factor	1
Setpoint [V]	0

Buttons: Insert, Return to menu

Figure 11: PID Controller Menu

This menu is for choosing the parameter of the PID controller (Fig. 11), which are the P-factor (proportional gain), the I-factor (integrate gain) and the D-factor (derivative gain), and the setpoint to which the system is supposed to be controlled. Since the input signal of the PIC controller is basically the output signal of the photodetector, the unit of the set point is voltage.

The parameters should be chosen as follows, otherwise the system can become unstable:

- $K_P = 0$

- $K_I = [0, 0.007]$
- $K_D = [0.06, 0.1]$

Controlling Menu

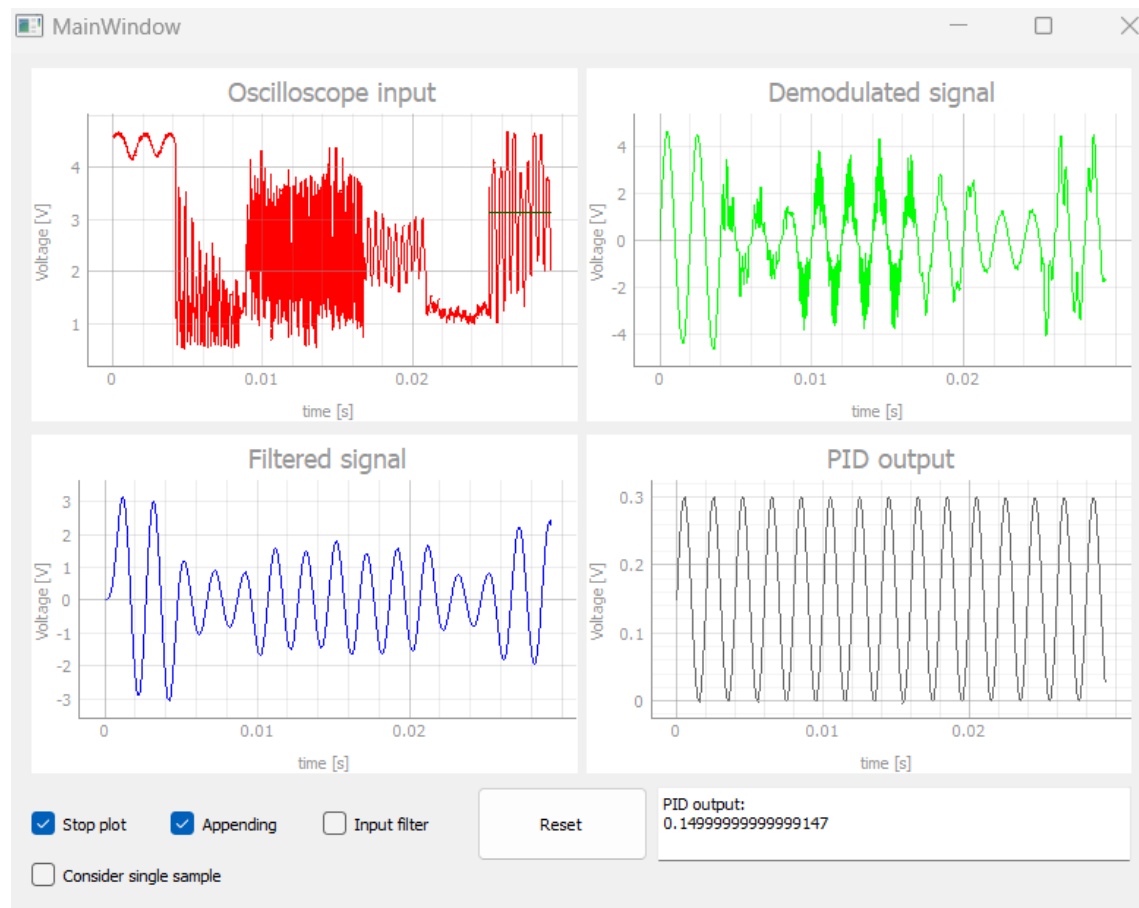


Figure 12: Controlling Menu

By clicking on the “start controlling” button in the main menu, another window opens which shows the measured and calculated signals in real-time (Fig. 12). There are 4 signals shown:

- Output voltage of the photodetector
- Demodulated signal
- Filtered signal
- PID output with the added dither signal

As soon as the window is opened, the control loop starts. Each cycle, the Moku:Go measures 1024 samples of the output voltage of the photodetector. Implicitly, all samples are appended and stored in a buffer up to 30720 points. When the 30720 samples are reached, the oldest 1024 samples, i.e. the samples at the beginning of the buffer, are deleted and the newly measured samples are appended at the end of the buffer stream. The reasons for limiting the buffer stream to 30720 samples is to decrease the computational time.

In the controlling menu, the user can choose between several options which influence the controlling itself but also the depiction of the signals

- Stop plot: If the tick is put, the real-time depiction stops, and the current signals can be considered
- Appending: If the tick is not put, only the latest measured samples are shown which are the latest 1024 samples. Otherwise, the monitor shows the whole sample stream which is stored in the buffer. By default, the tick is put.
- Input filter: If the tick is put, a low pass filter is applied to the input signal of the oscilloscope (i.e. output voltage of the photodetector). This filter is supposed to remove all higher frequencies above the dither frequency and therefore reduce the noise. The drawback of using the filter is the transient of the filter which is always depicted when the appending tick is put.
- Consider single sample: If the tick is put, the PID controller only considers the latest measured sample as input sample. Otherwise, if the tick is not set, the PID controller uses the mean value of all 1024 latest measured samples as input value. This setting was only introduced to compare the controlling by using the average of all measured 1024 samples per cycle and by just using the last of the 1024 samples. Comparing these two approaches showed that using the average leads to slightly better results.
- Reset button: If this button is pressed, the buffer is emptied and the PID controller is reset. This button should be used when a new measurement wants to be done.

- Plain text edit: The plain text edit on the right side of the menu prints the current output value of the PID controller excluding the dither signal.

Additionally, the graph of the oscilloscope input shows a green horizontal line. This line indicates the mean value of the last measured 1024 input samples. Furthermore, the output value of the PID controller is limited between the interval $[0, 10]$. This is necessary since the piezo crystal of the PZT-mirror would be damaged if negative voltage or voltage above 150V is applied. Since the piezoelectric controller amplifies its input signal by a factor of 15, the output voltage of the PID controller, which is also the input voltage of the piezoelectric controller, is not allowed to be higher than 10V.

In the monitor screen, the data can also be exported as csv-file or png-image. This can be done by right-clicking on the plot, which wants to be exported, and selecting “Export”. A sub-menu opens where the user can choose the format of the exporting data. During the real-time plot, the parameters of the other sub-menus can still be changed by just inserting new values. It is recommended to reset the plots afterwards.

Note: As well for exporting the real-time data as for changing the parameter values of the sub-menus, it is strongly recommended to stop the real-time plot for doing that. Otherwise, the program could become “laggy”.

2 Problems and Solutions

Problem 1

The fringes of the fringe pattern were too small compared to the lens of the detector. This means that it was not possible to receive a “complete” destructive interference pattern on the detector’s lens. So, the setpoint 0 was not able to be achieved.

The compromise is to choose another value as setpoint instead of 0. In order to choose a proper setpoint value the system must be fed with a ramp signal. This shows the boundary values of the output signal. Finally, any value between the boundary values can be used as setpoint.

Problem 2

Since the driver of the PZT-controller is only runnable on a 32-bit system but usually all PCs run a 64-bit Python version, the virtual environment, in which the whole Python program is run, must be set to a 32-bit environment. This can be done by opening the conda command line and inserting `set CONDA_FORCE_32BIT=1`. Afterwards, a

new virtual environment can be created as usual. After closing the conda command line window, conda is automatically reset to the 64-bit version again. However, before closing the cmd window, all packages should be installed. Otherwise conda will switch to its 64-bit version again which can cause problems when installing python packages.

Problem 3

For some windows systems, the API has got problems accessing the Moku:Go by using USB because the windows systems denies to connect. The solution for this problem is to configure a proxy which enables the system to listen to an IPv4 address and forward it to an IPv6 one. The steps are as follows:

- Open the cmd as administrator
- Insert: **netsh interface portproxy set v4tov6 listenport=8090 connectaddress=[fe80:0000:0000:0000:7269:79ff:feb9:0c22%3] connectport=http** where 8090 is the IPv4 address (can be chosen differently) and fe80:0000:0000:0000:7269:79ff:feb9:0c22%3 is the IPv6 address of the Moku:Go which is provided in the app. After setting up the proxy, the IPv4 address must be used to connect to Moku:Go in the API instead of the IPv6 address.

Problem 4

The controlling speed of the system is too slow. Since during each controlling cycle 1024 samples are measured and the average of all samples is used as input value of the PID controller, the system does not response to each single input sample but to each 1024 input samples. This leads to a significant delay. This problem has not been solved yet since the Moku:Go API does not provide a real time function for the oscilloscope yet. The controlling still works but it can be observed that sudden changes of the signal cannot be eliminated immediately by the controlling system.

Problem 5

Another problem is that due to the measurement of 1024 samples each cycle and calculating the average always removes the dither signal from the measured signal. This means, the whole idea of obtaining the first derivative of the intensity is negated. So, the system is then only controlled to the plain output voltage of the photodetector. However, in chapter 3 it can be seen that this approach still works. Still, the implemented

demodulation and low pass filter in the software are not completely useless. As soon as the Moku:Go API provides real-time data measurement, the software can be extended. Then the approach of using the first derivative of the intensity as input signal can be finalized.

3 Results

In the following section, the results of two different approaches are shown. The first approach is using the app and the PID controller which the Moku:Go provides. The second approach is using the software respectively the Python API of the Moku:Go.

3.1 Controlling by using the App

The Moku:Go already provides a user-interface and also a PID controller as instrument. If the dither signal is not be applied and the controlling is just be done according to the plain output of the photodetector, the PID controller of the Moku:Go can be used. Since the PID controller of the Moku:Go considers the signal before the transfer function as error signal and thus tries to bring this signal to 0, a negative offset value can be applied before the transfer function of the filter in order to set the setpoint (Fig. 13).

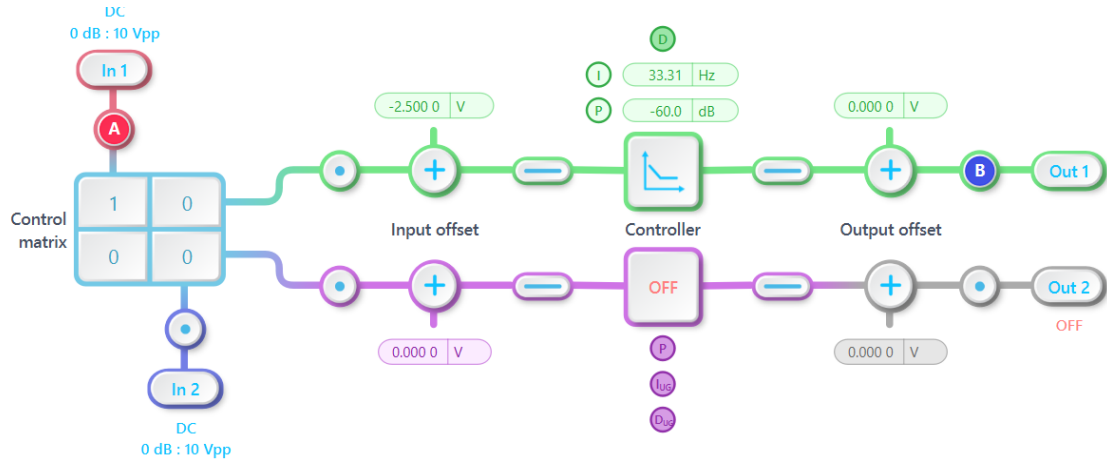


Figure 13: Setting setpoint to 2.5V

In order to get a suitable setpoint, the easiest way is to apply a ramp signal on the PZT controller and observe the output signal of the detector on the oscilloscope. The

oscilloscope shows the boundary values of the output signal and thus the setpoint value can be chosen as one value between the boundary values.

The main difficulty is finding proper parameter values for the PID parameter K_P , K_I and K_D . For getting sufficient parameter values, the following procedure can be done. First of all, the PID controller is just used as a P controller which means that the parameters K_I and K_D are turned off. The parameter K_P is set to 0dB. This is the starting point from where proper parameter values can be found.

Next, the value of the parameter K_P is increased. As long as there are strong oscillations in the input signal, the parameter value is too high (Fig. 14). So, from this point the parameter value is decreased as long as the strong oscillations disappear.

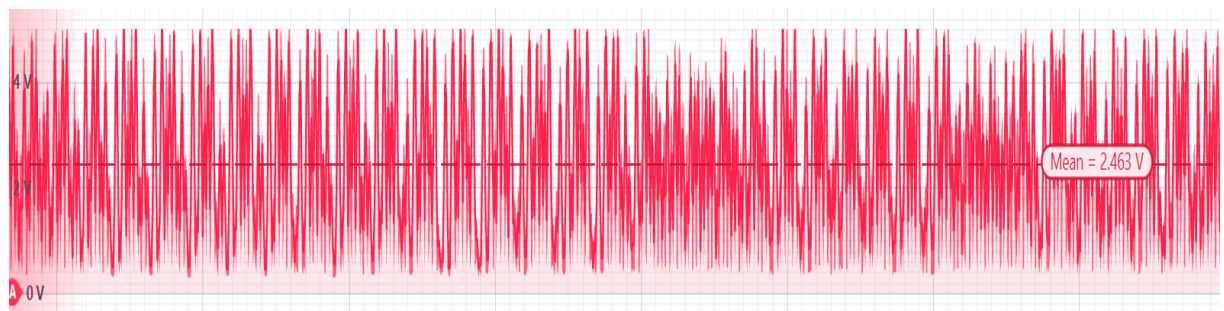


Figure 14: Oscillations due to too high value for K_P , setpoint = 2.5V

Subsequently, the parameter K_I is turned on. If the oscillations in the input signal are too strong, the parameter value must be decreased (Fig. 15).



Figure 15: Oscillations due to too high value for K_I , setpoint = 2.5V

If there are no significant oscillations but the controlling is not really accurate, the parameter value must be increased (Fig. 16).

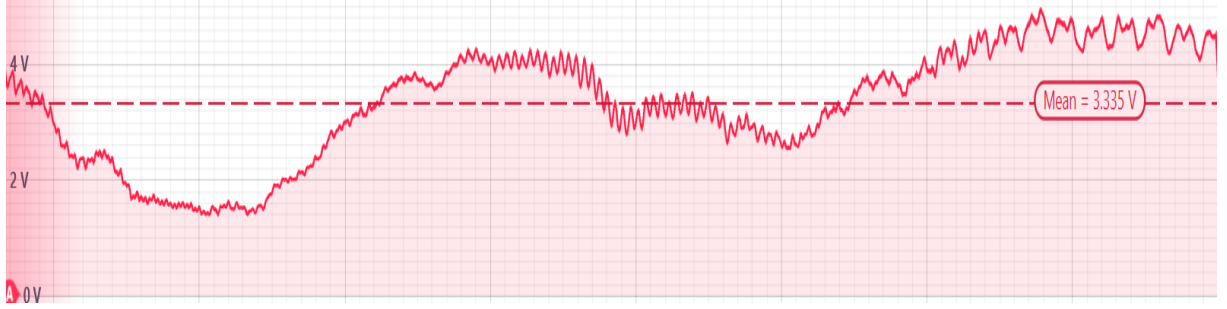


Figure 16: Worse controlling due to too low value for K_I , setpoint = 2.5V

The parameter K_D is neglected completely.

Results of the controlling applying quite proper parameter values can be seen in Fig. 17.

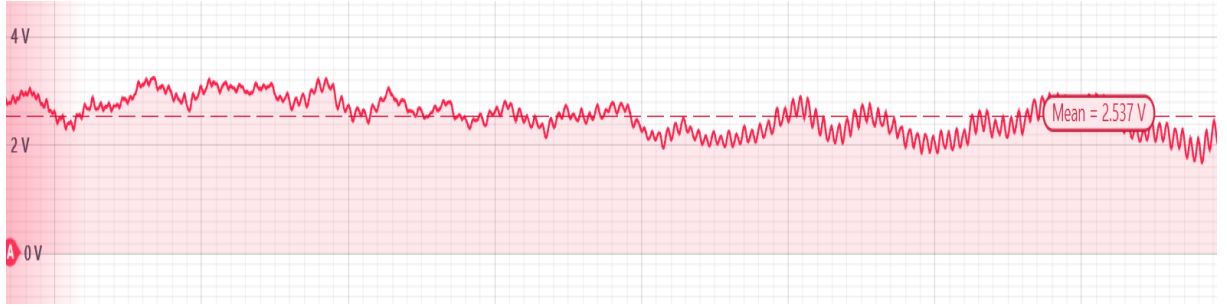


Figure 17: Controlling for proper parameter values, setpoint = 2.5V

3.2 Controlling by using the Software/Python API

As already mentioned, although the concept of using the first derivative of the intensity as controlling signal is implemented in the software, the concept can currently not be used since the Moku:Go does not provide real time data measurement for the oscilloscope yet. So, same as for the results by using the app, the results by using the software have also been obtained without using a dither signal. The controlling signal was just the plain output of the detector. In order to obtain trustworthy data, the setpoint was set to 2V (Fig. 18), 3V (Fig. 19) and 4V (Fig. 20).

For each setpoint it can be observed that the input signal of the PID controller generally oscillates around it. The reason why the signal does not look very smooth but stepped

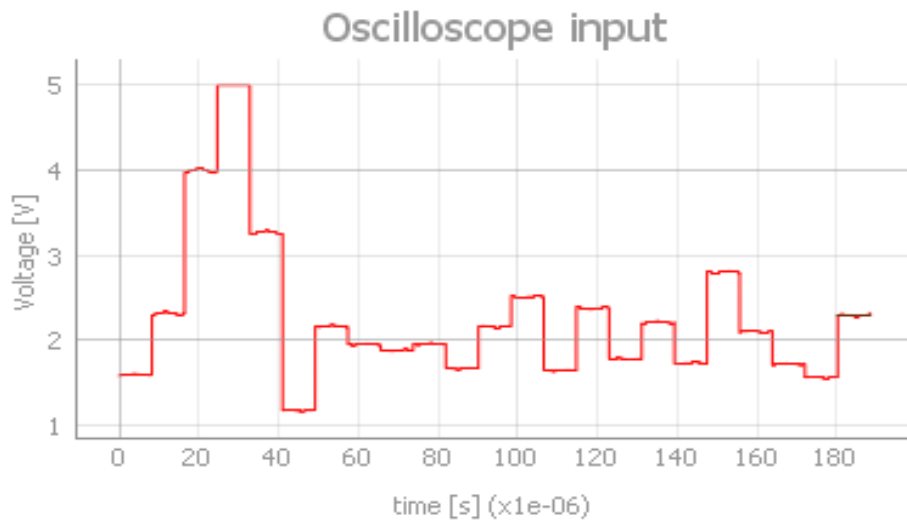


Figure 18: Controlling for setpoint = 2V

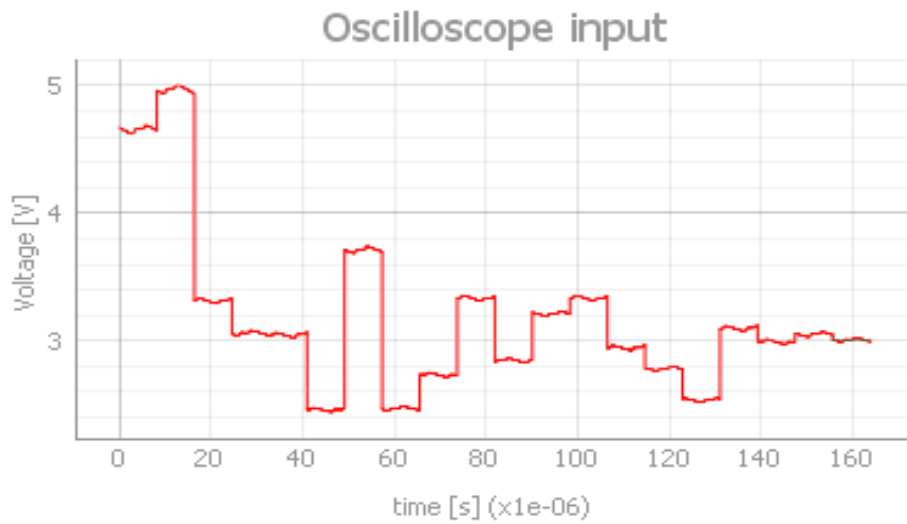


Figure 19: Controlling for setpoint = 3V

is that each step represents one measurement cycle of the oscilloscope, i.e., 1024 samples. Since for the controlling the average of all 1024 samples is used, each cycle the controller

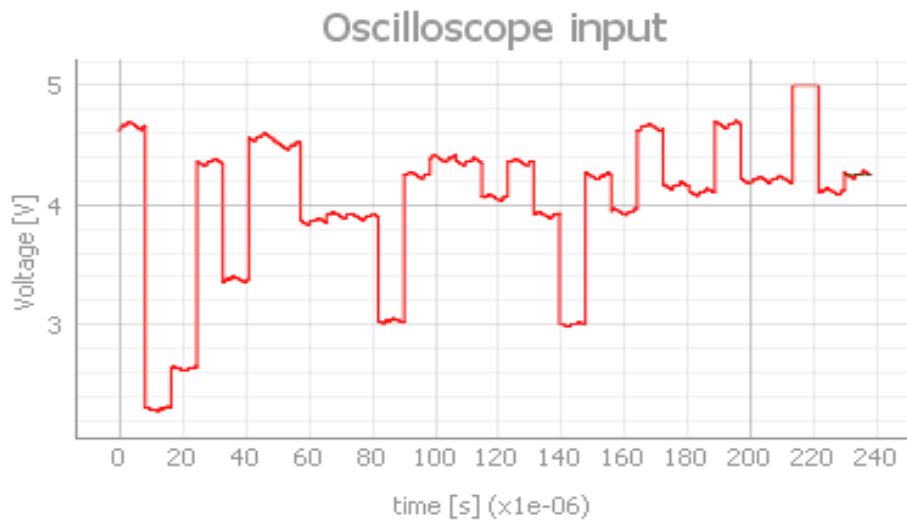


Figure 20: Controlling for setpoint = 4V

considers the 1024 samples as one sample. This shows clearly why the controlling is too slow and does not work perfect. Sometimes the system is not able to avoid runaways in the signal due to its limited controlling speed. However, it can also be clearly seen that the controlling generally works.

Bewertung des Praktikums

Separat zum eigentlichen Bericht, möchte ich noch eine kurze Praktikumsbewertung abgeben.

Das Praktikum bei TMOS war sehr schön. Die Verknüpfung der Bereiche der digitalen Signalverarbeitung, der physikalischen Optik, der Regelungstechnik und der Softwareentwicklung innerhalb des Projektes war sehr spannend. Die Leute sind sehr nett und man hat sehr viele Freiheiten in der Bearbeitung seines eigenen Projektes. Auch der Kontakt zu weltweit führenden Wissenschaftlern war sehr aufregend und natürlich hilft das Praktikum auch, die eigenen Englischkenntnisse zu verbessern.

Insgesamt ist ein Praktikum bei TMOS sehr zu empfehlen.