

Abschlussarbeit im Bachelorstudiengang Physik

**Modifikation des Versuches
'Schwingung und Resonanz' für das
physikalische Praktikum am WZW**

Andreas Grad

31. Juli 2019

Betreuer: R. Stoeppler
Erstgutachter(Themensteller): Prof. Dr. S. Paul
Zweitgutachter: Prof. Dr. R. Diehl

Inhaltsverzeichnis

Vorwort	vii
1 Schwingungsversuch	1
1.1 Theorie	1
1.1.1 Freie, ungedämpfte Schwingung	1
1.1.2 Freie, gedämpfte Schwingung	2
1.1.3 Erzwungene Schwingung	5
1.2 Der Versuchsaufbau	7
1.3 Notwendige Veränderungen und Rahmenbedingungen des Umbaus .	9
1.4 Analyse der bisherigen Genauigkeit	11
1.4.1 Genauigkeit der Eigenfrequenz	11
1.4.2 Messrate des Funkenschreibers	13
2 Entwicklung des neuen Versuches	15
2.1 Bestimmung der Messmethode	15
2.1.1 Beschleunigungssensoren	15
2.1.2 Magnetostruktive Sensoren	16
2.1.3 Magnetbandsensoren	16
2.1.4 Ultraschallsensoren	16
2.1.5 Optische Sensoren	17
2.1.6 Fazit	17
2.2 Auswahl des Sensors	17
2.2.1 Ultraschallsensor HC-SR04	17
2.2.1.1 Funktionsweise und Spezifikationen	17
2.2.1.2 Anschluss am Raspberry Pi	20
2.2.1.3 Messwiederholungsrate	23
2.2.1.4 Fokussierung des Messbereiches	23
2.2.1.5 Messgenauigkeit/Reproduzierbarkeit	26
2.2.1.6 Fazit	28
2.2.2 IR-Sensor Sharp GP2Y0A21YK0F	28
2.2.2.1 Funktionsweise und Spezifikationen	28
2.2.2.2 Anschluss am Raspberry Pi	31
2.2.2.3 Messwiederholungsrate	32
2.2.2.4 Messgenauigkeit/Reproduzierbarkeit	32

Inhaltsverzeichnis

2.2.2.5	Fazit	36
2.2.3	Ultraschallsensor UFA-1500-M18-A	37
2.2.3.1	Funktionsweise und Spezifikationen	37
2.2.3.2	Anschluss am Raspberry Pi	37
2.2.3.3	Messwiederholungsrate	38
2.2.3.4	Messgenauigkeit/Reproduzierbarkeit	38
2.2.3.5	Fazit	39
3	Bau des neuen Versuches	41
3.1	Hardware der Datenerfassung	41
3.2	Hardware der Datenverarbeitung	43
3.3	Software der Datenerfassung und Verarbeitung	45
3.4	Umbau der Bedienung des Motors	47
3.4.1	Umbau der Motorsteuerung	47
3.4.2	Umbau der Frequenzbestimmung	48
4	Überlegungen zur Didaktik	53
A	Abkürzungsverzeichnis	57
B	Quellcode	59
B.1	Klasse für den Analog Digital Wandler MCP3008	59
B.2	Ansteuerung des Ultraschallsensors HC-SR04	60
B.3	Auslesen des Infrarotsensors GP2Y0A21YK0F	62
B.4	Grafische Oberfläche für den Sensor	63
B.4.1	Startfenster	63
B.4.2	Fenster zum Aufnehmen der Messung	66
B.4.3	Fenster zum Speichern der freien Schwingungen	69
B.4.4	Fenster zum Speichern der erzwungenen Schwingungen	71
B.4.5	Fenster zum Kalibrieren des Sensors	73
B.4.6	Fenster zur Eingabe des Messbereiches	75
B.5	Grafische Oberfläche für die Motorsteuerung	77
C	Schaltpläne	79
C.1	Steuerung und Auswertung des Sensors	80
C.2	Frequenzbestimmung mit dem Generator	81
C.3	Remote Control Steuerung des Netzgerätes für den Motor	82
D	Bauteile	83
E	Gehäuse für den Raspberry Pi	87

Inhaltsverzeichnis

F Abbildungsverzeichnis	91
Literatur	95

Vorwort

Physik ist eine Wissenschaft, die sich mit den grundlegenden Phänomenen unserer Welt beschäftigt. Viele verschiedene Studiengänge bauen daher auf diesen Grundstock auf. Speziell am Wissenschaftszentrum Weihenstephan mit der Spezialisierung auf die Bereiche Ernährung, Landnutzung und Umwelt, hat die Physik eine große Bedeutung und ist oft zum Verständnis diverser Sachverhalte vonnöten. Aus diesem Grund spielt die Physikausbildung dort eine bedeutende Rolle. Mithilfe des physikalischen Praktikums soll den Studenten ein Gefühl für die Zusammenhänge in unserer Natur sowie die Bedeutung physikalischer Formeln und deren Anwendung in der Realität vermittelt werden. Deswegen wurden aus den Bereichen Mechanik, Wärme, Optik und Elektrizität 12 Praktikumsversuche für die Studenten dort geschaffen. Diese sollen den Studierenden neben der Vorlesung die Möglichkeit geben, sich mit diversen Teilgebieten der Physik einmal selbst zu befassen.[1]

Diese Versuchsaufbauten sind nun jedoch zum Teil in die Jahre gekommen, sodass nicht mehr alle funktionstüchtig sind. Im Rahmen dieser Arbeit soll speziell der Versuch 'Schwingung und Resonanz' wieder instand gesetzt werden. Dabei soll der aktuelle Stand der Technik genutzt werden, um die Versuchsdurchführung unkomplizierter und anschaulicher zu gestalten. Zusätzlich soll jedoch auch Wert auf die Didaktik der Physik gelegt werden, um die Studenten selbst zu fordern und ihnen diesen Teilbereich der Physik näher zu bringen.

Kapitel 1

Schwingungsversuch

1.1 Theorie

1.1.1 Freie, ungedämpfte Schwingung

Wird ein Körper, welcher mit einer Feder verbunden ist, um eine bestimmte Strecke x bewegt, so wirkt auf diesen laut dem Hook'schen Gesetz die Kraft F

$$F = -D \cdot x \quad (1.1)$$

Stellt man hierfür nun die Bewegungsgleichung auf, folgt daraus:

$$m \cdot \frac{d^2x}{dt^2} = -D \cdot x \quad (1.2)$$

Dies ist eine Differentialgleichung zweiter Ordnung, welche sich mit dem Ansatz

$$x(t) = A \cos(\omega_0 \cdot t) \quad (1.3)$$

lösen lässt. Der Faktor A ist Amplitude der Schwingung und lässt sich aus den Anfangsbedingungen bestimmen. Die Eigenkreisfrequenz dieser Schwingung, also ω_0 , lässt sich durch Einsetzen des Ansatzes in die gegebene Differentialgleichung bestimmen und ist in diesem Fall gleich der Wurzel aus dem Verhältnis von Federkonstante und Masse:

$$\omega_0 = \sqrt{\frac{D}{m}} \quad (1.4)$$

Aus der Eigenfrequenz lässt sich wiederum die Schwingungsdauer T und Frequenz f mithilfe folgender Zusammenhänge herleiten:

$$T = \frac{2\pi}{\omega_0} \text{ und } f = \frac{1}{T} = \frac{\omega_0}{2\pi} \quad (1.5)$$

Abbildung 1.1 zeigt den Verlauf einer ungedämpften Schwingung mit einer Amplitude von $4m$ und einer Eigenfrequenz ω_0 von $1.5\frac{1}{s}$:

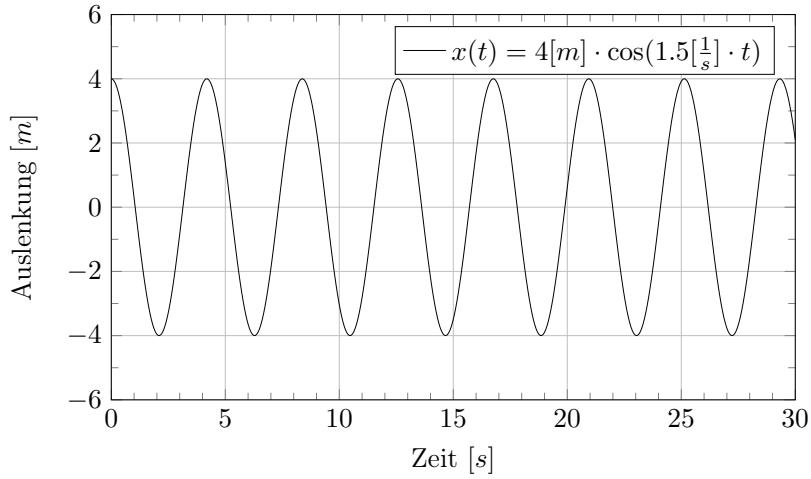


Abbildung 1.1: Ungedämpfte Schwingung

[1]

1.1.2 Freie, gedämpfte Schwingung

Eine ideale, freie, ungedämpfte Schwingung ist natürlich in der Praxis nicht realisierbar. Es geht immer Energie in Form von Reibung verloren. Dadurch wird die Energie in unserem System verkleinert und die maximale Auslenkung, sprich die Amplitude der Schwingung, verringert sich im Laufe der Zeit. Man spricht nun von einer gedämpften Schwingung, da die Dämpfung der Schwingung entgegen wirkt. Bei mechanischen Schwingungen, wie in diesem Versuch, ist diese proportional zur Geschwindigkeit v , also proportional zu $\frac{dx}{dt}$. Daraus ergibt sich ein weiterer Summand in unserer Differentialgleichung, der nun berücksichtigt werden muss.

$$m \cdot \frac{d^2x}{dt^2} = -D \cdot x - k \cdot \frac{dx}{dt} \quad (1.6)$$

Zum Lösen dieser Differentialgleichung werden zuerst zwei neue Variablen eingeführt, welche die Rechnung im Anschluss vereinfachen werden. Die Variable ω_0 wird hierbei als Eigenfrequenz des Systems bezeichnet und die Variable δ als Dämpfung:

$$\omega_0 = \sqrt{\frac{D}{m}} \text{ und } \delta = \frac{k}{2m} \quad (1.7)$$

Daraus ergibt sich dann folgende Gleichung:

$$\ddot{x} + \omega_0^2 x + 2\delta\dot{x} = 0 \quad (1.8)$$

Nun wird der Exponentialansatz $x(t) = A \cdot \exp(\lambda t)$ verwendet und es folgt eine quadratische Gleichung für die Parameter λ :

$$\lambda_{1,2} = -\delta \pm \sqrt{\delta^2 - \omega_0^2} \quad (1.9)$$

Die allgemeine Lösung der Differentialgleichung ergibt sich aus der Linearkombination der Lösungen mit den beiden Parametern λ_1 und λ_2 :

$$x(t) = A \cdot \exp(-\delta t) \cdot \left(\exp(\sqrt{\delta^2 - \omega_0^2}t) + \exp(-\sqrt{\delta^2 - \omega_0^2}t) \right) \quad (1.10)$$

Hierbei unterscheidet man nun zwischen 3 verschiedenen Fällen:

- Schwingfall: Hier ist die Dämpfung relativ klein und es gilt der Zusammenhang $\delta^2 - \omega_0^2 < 0$. Das System übt in diesem Fall immer noch eine Schwingung aus, auch wenn die Amplitude der Schwingung immer kleiner wird.
- Aperiodischer Grenzfall: In diesem Fall herrscht zwischen der Dämpfung und der Eigenfrequenz der Zusammenhang $\delta^2 - \omega_0^2 = 0$. Das System übt hierbei nur eine einzige Schwingung aus und kehrt danach in schnellstmöglicher Zeit in den Ruhezustand zurück.
- Kriechfall: Hier ist die Dämpfung größer als die Eigenfrequenz des Systems und es gilt die Gleichung: $\delta^2 - \omega_0^2 > 0$. Dieser Fall ist ähnlich zum aperiodischen Grenzfall. Der einzige Unterschied ist, dass das System nicht in schnellstmöglicher Zeit in die Ruhelage zurückkehrt.

Da für diesen Versuch nur der Schwingfall von Interesse ist, also der Fall $\delta^2 - \omega_0^2 < 0$, lässt sich Gleichung 1.10 mithilfe der Euler Formel:

$$\cos(x) = \frac{e^{ix} + e^{-ix}}{2} \quad (1.11)$$

noch folgendermaßen vereinfachen:

$$x(t) = 2A \cdot \exp(-\delta t) \cdot \cos(\sqrt{\omega_0^2 - \delta^2}t) \quad (1.12)$$

Diese Schwingung hat nun eine maximale Amplitude von $2A$ und besteht aus dem Produkt einer Schwingung mit der Frequenz $\omega = \sqrt{\omega_0^2 - \delta^2}$ und einer abklingenden Exponentialfunktion. Die Frequenz dieser Schwingung ist somit, wie man an der Formel erkennen kann, kleiner als die der ungedämpften Schwingung. Das heißt,

die Dämpfung verringert nicht nur die maximale Auslenkung mit der Zeit, sondern erhöht auch die Periodendauer einer Schwingung ganz allgemein. Ein Beispiel für eine solche Schwingung ist in folgender Grafik mit der Anfangsbedingung $x(0) = 4m$ aufgetragen.

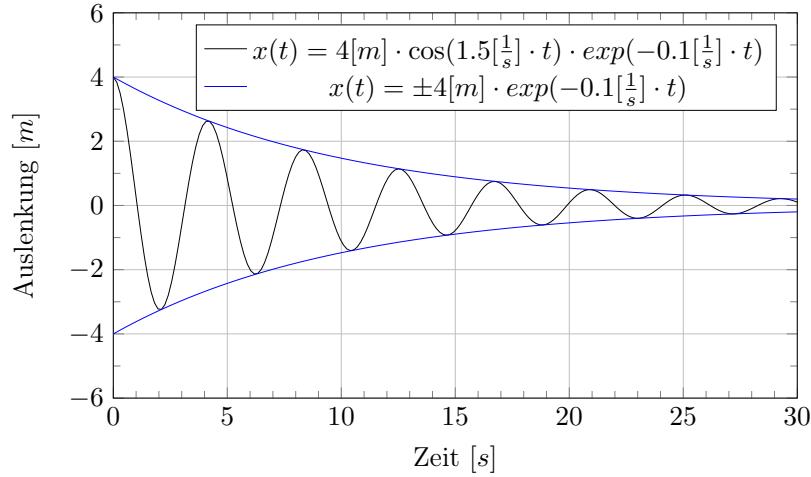


Abbildung 1.2: Gedämpfte Schwingung

Wie man sehr schön erkennen kann, wird die Schwingung sowohl von oben als auch von unten durch die abklingende Exponentialfunktion eingeschlossen. Das heißt, die Amplitude der Schwingung wird immer kleiner, wird in der Theorie jedoch niemals null erreichen. Dies ist aber natürlich in der Praxis nicht durchführbar, da die Auslenkung ab einer bestimmten Zeit einfach nicht mehr messbar ist. Das Besondere hierbei ist, dass aufgrund der Periodizität der Cosinus-Funktion das Verhältnis zweier aufeinander folgender Maxima nur von der Exponentialfunktion abhängt:

$$q = \frac{\text{Maximum}_{n-1}}{\text{Maximum}_n} = \frac{2A \cdot e^{-\delta t} \cdot \cos(\omega t)}{2A \cdot e^{-\delta(t+T)} \cdot \cos(\omega(t+T))} = e^{\delta T}, \quad (1.13)$$

wobei T die Periodendauer einer Schwingung ist. Durch Logarithmierung erhält man daraufhin einen Ausdruck für das logarithmische Dekrement λ :

$$\lambda = \ln(q) = \delta \cdot T \quad (1.14)$$

Das logarithmische Dekrement ist ein Maß für das Dämpfungsverhalten einer Schwingung. Es ist immer positiv und je größer der Wert, desto größer auch die Dämpfung eines Systems. [1]

1.1.3 Erzwungene Schwingung

Es nicht nur möglich, einer Schwingung mittels Reibung etc. Energie zu entziehen, man kann ihr auch Energie zuführen. Dies geschieht meist durch eine periodische Kraft, welche auf den schwingenden Körper einwirkt.

$$F = F_0 \cdot \cos(\omega_e t) \quad (1.15)$$

Diese Kraft hat einen Maximalwert von F_0 und oszilliert mit der Frequenz ω_e . Dies bewirkt wiederum einen neuen Term in der bereits vorhandenen Differentialgleichung:

$$m \cdot \frac{d^2x}{dt^2} = -D \cdot x - k \cdot \frac{dx}{dt} + F_0 \cdot \cos(\omega_e t) \quad (1.16)$$

Da das explizite Lösen dieser Gleichung etwas aufwändiger ist, wird darauf im Rahmen dieser Arbeit nicht weiter eingegangen. Eine spezielle Lösung dieser Gleichung ist jedoch folgender Ansatz:

$$x(t) = A \cos(\omega_e t - \phi) \quad (1.17)$$

Das heißt, nach dem Einschwingvorgang schwingt das System mit derselben Frequenz wie die externe Kraft selbst, also ω_e . Der Einschwingvorgang bezeichnet die Zeit, die das System benötigt, um sich an die externe Kraft anzupassen. Dabei ist die Schwingung jedoch noch um den Phasenunterschied ϕ zu der ursprünglichen Schwingung der Kraft verschoben. Dieser Phasenunterschied lässt sich wie folgt berechnen:

$$\phi = \arctan\left(\frac{k\omega_e}{m(\omega_0^2 - \omega_e^2)}\right) \quad (1.18)$$

Für die Amplitude A der erzwungenen Schwingung ergibt sich der zeitlich konstante Ausdruck:

$$A = \frac{F_0}{\sqrt{m^2(\omega_0^2 - \omega_e^2)^2 + k^2\omega_e^2}}, \quad (1.19)$$

welcher sich mit $\omega_0 = \sqrt{\frac{D}{m}}$ und $\delta = \frac{k}{2m}$ noch vereinfachen lässt:

$$A = \frac{\frac{F_0}{m}}{\sqrt{(\omega_0^2 - \omega_e^2)^2 + 4\delta^2\omega_e^2}} \quad (1.20)$$

Trägt man nun die Auslenkung A in Abhängigkeit der Erregerfrequenz ω_e auf, erhält man die sogenannte Resonanzkurve, welche in folgender Abbildung zu sehen ist. Die verwendeten Parameter sind zur besseren Übersichtlichkeit separat in der Tabelle zu dargestellt.

	blau	rot	grün
$F_0[N]$	6	2	1
$m[kg]$	1	1	1
$\omega_0[\frac{1}{s}]$	6.7	6	7
$\delta[\frac{1}{s}]$	0.5	0.4	0.8

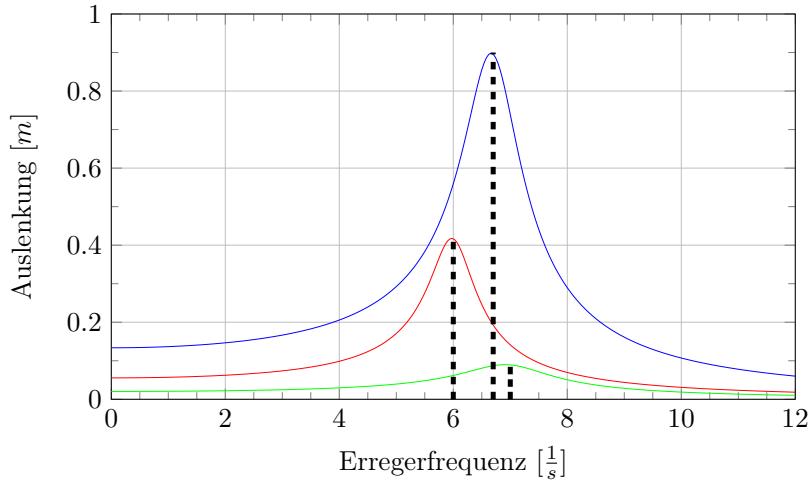


Abbildung 1.3: Erzwungene Schwingung

Das Besondere hierbei ist, dass die Amplitude maximal ist, sobald die Erregerfrequenz gleich der Eigenfrequenz des Schwingers, also ω_0 ist. Das bezeichnet man auch als Resonanzfall, das heißt, dass genau dort die Erregerfrequenz der Eigenfrequenz des Systems entspricht. Stimmen diese beiden Frequenzen überein, ist es möglich, mit einer nur minimalen Erregeramplitude eine riesige Schwingungsamplitude zu erzeugen. Dies hat besonders in der technischen Anwendung eine große Bedeutung, da Materialien durch diese immense Schwingung sehr großen Schaden nehmen können. Deshalb muss die Besonderheit des Resonanzfalles in nahezu jedem Bereich, vom Brückenbau bis hin zur Konstruktion von den Tragflächen für Flugzeuge, berücksichtigt werden.

Ein weiteres Merkmal dieser Kurve ist ihre Bandbreite b . Die Bandbreite bezeichnet das Intervall, in dem die Amplitude gleich oder größer als das $\frac{1}{\sqrt{2}}$ -fache der maximalen Amplitude ist. Die Grenzen dieses Intervalls sind somit $\omega_{e1,2}$ mit :

$$A(\omega_{e1,2}) = \frac{A_{max}}{\sqrt{2}} \quad (1.21)$$

Ist die Dämpfung eines Systems relativ klein, gilt hierfür folgende Beziehung zwischen Bandbreite und Dämpfung:

$$\delta = \frac{b}{2} \quad (1.22)$$

[1]

Die Bandbreite der Resonanzkurve ist in folgender Abbildung noch einmal grafisch dargestellt. Die Resonanzfrequenz ist hierbei bei $0.59\frac{1}{s}$ und ihre Amplitude, also das Maximum, bei $0.59m$. Daraus folgt für die Grenzen ein Wert von:

$$A(\omega_{e1,2}) = \frac{0.59m}{\sqrt{2}} = 0.42m \quad (1.23)$$

$$\Rightarrow \omega_{e1} = 5.1\frac{1}{s} \quad (1.24)$$

$$\Rightarrow \omega_{e2} = 6.6\frac{1}{s} \quad (1.25)$$

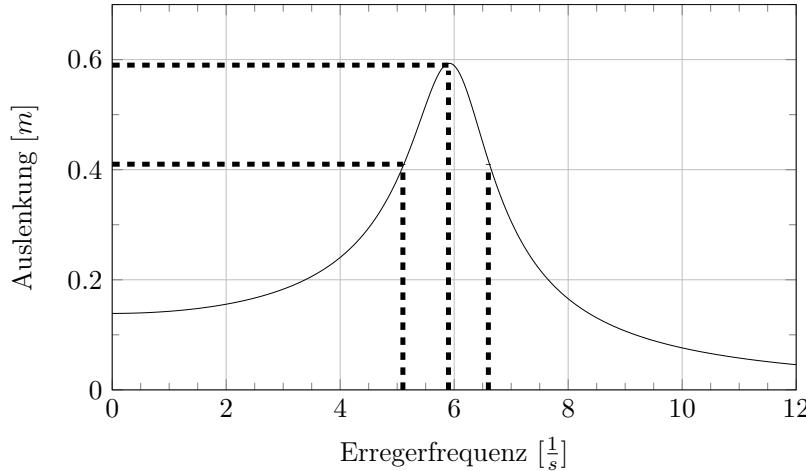


Abbildung 1.4: Halbbreite der Resonanzkurve

1.2 Der Versuchsaufbau

Im Folgenden soll der bisherige Versuchsaufbau und die einzelnen Komponenten kurz dargestellt werden.

Kapitel 1 Schwingungsversuch

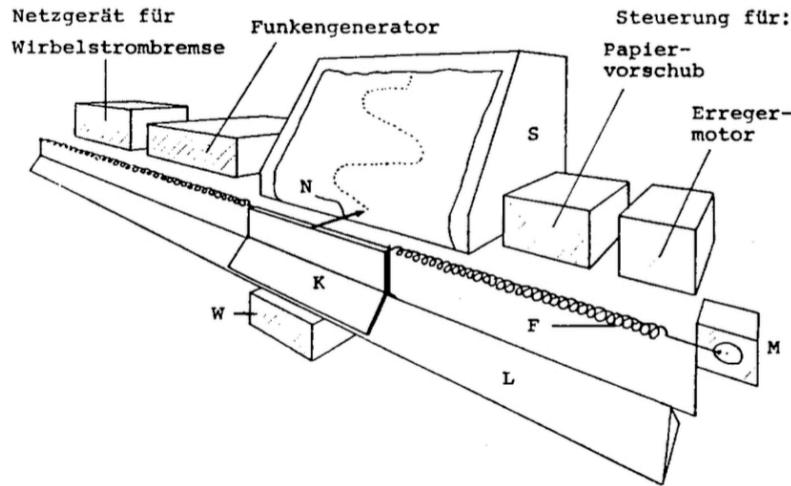


Abbildung 1.5: Versuchsaufbau[1]

Die Schwingung an sich wird durch einen Gleiter (K) visualisiert, welcher sich auf einer gebogenen Metallschiene bewegt und an beiden Enden mit einer Feder (F) verbunden ist. Um die Reibung zu verringern wurden in die Schiene Löcher gebohrt, sodass daraus eine Luftkissenfahrbahn (L) entsteht.

Erzwungene Schwingungen können durch den Erregermotor (M), welcher an einer der beiden Federn befestigt ist, realisiert werden. Dieser wird durch einen variablen Gleichstrom von $1 - 6V$ gesteuert und erreicht hierbei eine Frequenz ω_e von $0.2\frac{1}{s}$ bis etwa maximal $1.8\frac{1}{s}$. Die Spannung für diesen wird über ein Potentiometer gesteuert. An die Antriebswelle ist außerdem ein Generator angeschlossen, welcher auf Basis der tatsächlichen Motorfrequenz eine Spannung ausgibt, die dann an einem Display angezeigt wird.

Um den Studierenden die Möglichkeit zu geben, Messungen mit verschiedenen Dämpfungen aufnehmen zu können, ist außerdem eine Wirbelstrombremse an der Luftkissenbahn montiert. Durch ein Netzgerät, mit dem die Stärke der Dämpfung reguliert werden kann, wird den Studenten ein Gefühl gegeben, welchen Einfluss Reibung auf eine eigentlich fast ideale Schwingung haben kann. Der Aufbau dieser Bremse ist in folgender Grafik schematisch abgebildet:

1.3 Notwendige Veränderungen und Rahmenbedingungen des Umbaus

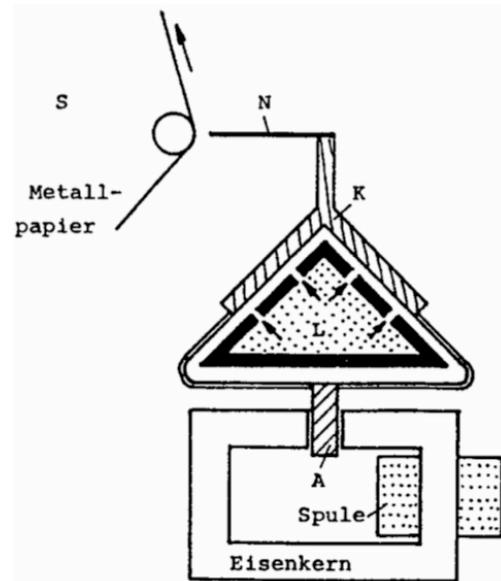


Abbildung 1.6: Aufbau der Wirbelstrombremse[1]

Das Aufzeichnen der Schwingung erfolgt durch einen Funkenschreiber (S). Hierbei werden durch den Funkengenerator zwischen der Nadel (N) und dem Metallpapier Hochspannungsfunken erzeugt. Die Funken brennen beim Auftreffen einen schwarzen Punkt in das Papier. Da das Papier durch den Papiervorschub mit einer konstanten Geschwindigkeit bewegt werden kann, ist im Nachhinein anhand der Punkte die Schwingungskurve auf dem Blatt sehr gut zu erkennen.

Die Frequenz der Funken und die Geschwindigkeit des Papiervorschubes lassen sich beide manuell einstellen und sollten immer an die jeweilige Messung angepasst werden. Ist die Bewegung des Gleiters beispielsweise durch die Dämpfung sehr langsam, sollte auch eine geringere Frequenz für die Funken eingestellt werden. Ansonsten wäre die Schwingungskurve auf dem Papier anhand viel zu vieler Brennpunkte sehr dick und die Ungenauigkeit würde dadurch zunehmen.[1]

1.3 Notwendige Veränderungen und Rahmenbedingungen des Umbaus

Da die Versuchsanordnung bereits im Jahre 1982 gebaut wurde, ist die Methode zur Messungsaufnahme selbstverständlich nicht mehr auf dem aktuellsten Stand der Technik. Dies wäre an sich kein Problem, doch mittlerweile wird das für den Funkenschreiber benötigte Metallpapier nicht mehr produziert. Da ohne dieses Papier

Kapitel 1 Schwingungsversuch

die Aufnahme der Messkurven nicht möglich ist, soll im Rahmen dieser Arbeit eine neue Methode entwickelt werden, die Messdaten zu erfassen.

Außerdem ist bereits bei zwei der sechs Versuchsaufbauten die Anzeige zum Ablesen der Motorfrequenz nicht mehr funktionsfähig. Hier wurde als Übergangslösung ein Spannungsmessgerät an den Generator angeschlossen. Die Frequenz konnte daraufhin anhand eines Plots, in dem die Motorfrequenz in Abhängigkeit der Spannung aufgetragen ist, bestimmt werden. Da in Zukunft auch mit dem Ausfall von weiteren Anzeigen zu rechnen ist, wurde beschlossen, während des Umbaus auch gleich die Motorsteuerung und Anzeige der Frequenz auszutauschen.

Alle vorgenannten Änderungen sollten an einem der Aufbauten umgesetzt werden. Als Basis des neuen Versuchs sollte der Einplatinencomputer Raspberry Pi verwendet werden. Vor Beginn der Arbeit wurden folgende Rahmenbedingungen festgelegt, welche bei dem Umbau zu beachten sind:

- Der Versuchsaufbau ist für Studenten mehrerer Studienrichtungen ausgelegt und wird daher sehr oft durchgeführt werden. Aus diesem Grund muss der gesamte Aufbau stabil, robust und kompakt gestaltet werden, um eine möglichst hohe Nutzungsdauer zu gewährleisten.
- Nachdem die meisten Studenten das Physikpraktikum in einem der ersten Semester haben, sollte die Bedienung des Versuches relativ einfach und selbsterklärend umgesetzt werden. Der Fokus liegt hier auf dem Verständnis der physikalischen Grundlagen und sollte nicht durch eine komplizierte und mühsame Bedienung erschwert werden.
- Da der Versuch erwartungsgemäß stark beansprucht wird, sollten nach Möglichkeit defekte Einzelteile möglichst schnell austauschbar sein, um einen fließenden Praktikumsbetrieb zu gewährleisten.
- Durch den Umbau sollte die Genauigkeit und Reproduzierbarkeit der Messdatenerfassung nicht verschlechtert werden.
- Außerdem soll bei jeder Änderung der Einfluss auf die Didaktik der Physik, die hinter dem Versuch steht, beachtet werden. Es sollte einerseits zwar alles möglichst einfach und unkompliziert durchführbar sein, jedoch trotzdem auch der Fokus auf das Verständnis und die Hintergründe der einzelnen Messungen gelegt werden. Das heißt, die Studenten sollen dazu angeregt werden, die Messtechnik, das Prinzip dahinter und den Grund der einzelnen verschiedenen Messungen zu hinterfragen und zu verstehen.

1.4 Analyse der bisherigen Genauigkeit

1.4.1 Genauigkeit der Eigenfrequenz

Um die Genauigkeit dieses Versuches zu bewerten, wurde die Eigenfrequenz des Gleiters auf der Luftkissenbahn sowohl theoretisch als auch experimentell bestimmt. Mögliche Einflüsse, die diese Messung verfälschen können sind:

- Reibung zwischen Gleiter und Bahn trotz Luftkissenbahn
- Reibung durch die Bewegung und den Luftwiderstand
- Ungenauigkeit beim Auftragen der Schwingung durch den Funkenschreiber
- Messungenauigkeit beim Abmessen der Periode der Schwingung auf dem Papier

Um diesen Wert dennoch möglichst genau bestimmen zu können, wurde die Messung für die freie, ungedämpfte Schwingung drei mal aufgezeichnet. Außerdem wurden bei jeder Messung mindestens 15 Schwingungen aufgezeichnet, um den Fehler beim Abmessen der Periode zu minimieren.

Mögliche Fehler bei der theoretischen Berechnung sind:

- Messungenauigkeit bei der Bestimmung der Masse des Gleiters
- Inhomogenität der Federkonstante bei den beiden Federn

Diese Werte sind jedoch bereits Versuchs spezifisch angegeben und werden nun als $m = 0.815\text{kg}$ und $D = 2.780\frac{\text{N}}{\text{m}}$ angenommen.[1] Der theoretische Wert lässt sich nach Gleichung 1.7 wie folgt berechnen:

$$\omega_0 = \sqrt{\frac{D}{m}} = \sqrt{\frac{2.780\frac{\text{N}}{\text{m}}}{0.815\text{kg}}} = 1.847\frac{1}{\text{s}} \quad (1.26)$$

$$f_0 = \frac{\omega_0}{2\pi} = 0.294\text{Hz} \quad (1.27)$$

Für die experimentelle Bestimmung sind die Messwerte in folgender Tabelle angegeben:

Messung	1	2	3
Anzahl der Schwingungen n	17	19	22
Schreibgeschwindigkeit v	$5\frac{\text{mm}}{\text{s}}$	$5\frac{\text{mm}}{\text{s}}$	$5\frac{\text{mm}}{\text{s}}$
Distanz D	29.2cm	32.4cm	37.7cm

Die Schreibgeschwindigkeit hat hierbei eine Unsicherheit von $\pm 0.1 \frac{mm}{s}$ und für die Ungenauigkeit beim Abmessen der Distanz wurde $1mm$ angenommen. Mithilfe der Geschwindigkeit des Papiervorschubes v und der Gesamtlänge der Aufzeichnung D lässt sich die Gesamtzeit einer Messung t bestimmen:

$$t = \frac{D}{v} \quad (1.28)$$

Die Periodendauer einer Schwingung erhält man durch die Anzahl der Schwingungen n :

$$T = \frac{t}{n} \quad (1.29)$$

Die Frequenz der Schwingung lautet dann:

$$f_0 = \frac{1}{T} = \frac{n}{t} = \frac{n \cdot v}{D} \quad (1.30)$$

Für die drei Messungen ergeben sich dann folgende Werte:

Messung	Frequenz f_0
1	0.291Hz
2	0.293Hz
3	0.292Hz

Die Fortpflanzung der Unsicherheiten lässt sich mithilfe der Gauß'schen Fehlerfortpflanzung berechnen:

$$\Delta f_0 = \sqrt{\left(\frac{\Delta v}{v}\right)^2 + \left(\frac{\Delta D}{D}\right)^2} \cdot f_0 \quad (1.31)$$

Daraus folgt dann:

Messung	Frequenz f_0
1	$(0.291 \pm 0.006)Hz$
2	$(0.293 \pm 0.006)Hz$
3	$(0.292 \pm 0.006)Hz$

Somit ist der Mittelwert der experimentell bestimmten Eigenfrequenz gleich:

$$f_0 = (0.292 \pm 0.003)Hz \quad (1.32)$$

Dieses Messergebnis besitzt im Bezug auf den theoretischen Wert nur eine Abweichung von 0.7% und der theoretisch berechnete Wert liegt ausserdem innerhalb der Messungsgenauigkeit des experimentell bestimmten Wertes. Dies zeigt, dass der Versuch eine sehr gute Wahl ist, um den Studierenden eine freie, ungedämpfte

Schwingung zu zeigen und erläutern. Außerdem bedeutet dies auch, dass der Funkenschreiber eine sehr gute Möglichkeit zum Aufzeichnen der Schwingung war. Diese Genauigkeit sollte im Laufe des Umbaus auch erhalten bleiben.

1.4.2 Messrate des Funkenschreibers

Um die Messrate des Funkenschreibers abschätzen zu können, wurde die Frequenz der Funken auf die höchste Stufe gestellt und eine Schwingung mit möglichst großer Amplitude aufgenommen, um die Auflösung zu maximieren. Daraufhin wurde die Anzahl der Schwärzungen N auf dem Metallpapier in Abhängigkeit der Distanz D und der Schreibgeschwindigkeit v bestimmt. Die Frequenz der Funken ergibt sich somit aus:

$$f = \frac{v \cdot N}{D} \quad (1.33)$$

Die Messungenauigkeiten ergeben sich hierbei ebenfalls aus der Ungenauigkeit der Schreibgeschwindigkeit und beim Abmessen der Distanz D . Sie wurde daher wieder mit Gleichung 1.31 berechnet. Mit $N = 168$ bei einer Distanz von $D = 25.5\text{mm}$ folgt daraus:

$$f = (32.9 \pm 1.5)\text{Hz} \quad (1.34)$$

Diese Unsicherheit mit fast 5% ist relativ hoch und könnte durch eine längere Messdauer sowie einer größeren Anzahl an Messungen verbessert werden. Da diese Messung jedoch nur zur Abschätzung der Frequenz dient, ist das völlig ausreichend. Im Zuge des Umbaus sollte daraufhin versucht, wieder eine Messfrequenz in etwa diesem Bereich zu ermöglichen.

Kapitel 2

Entwicklung des neuen Versuches

2.1 Bestimmung der Messmethode

Um die Daten in Zukunft zu erfassen und dann elektronisch an den Raspberry Pi weiterzuleiten, kamen verschiedene Sensorarten in Frage:

- Beschleunigungssensoren
- Magnetostriktive Sensoren
- Magnetband Sensoren
- Ultraschallsensoren
- optische Sensoren

Die wichtigsten Punkte bei der Auswahl waren im ersten Schritt die Möglichkeit zur Montage und zur Stromversorgung der Sensoren und die Robustheit des gesamten Aufbaus.

2.1.1 Beschleunigungssensoren

Beschleunigungssensoren erfassen die Beschleunigung, die auf sie selbst wirkt. Da die Beschleunigung an den Maxima der Schwingung gleich null ist, könnte man dadurch die Periode einer Schwingung berechnen. Anhand der Beschleunigung selbst ist es außerdem möglich, die Ort-Zeit-Kurve der Schwingung zu berechnen. Da der Beschleunigungssensor auf dem Gleiter montiert werden muss, folgt daraus, dass entweder eine kleine Stromversorgung zusätzlich auf dem Gleiter montiert werden muss oder die Versorgung extern über ein Kabel erfolgt. Die Stromversorgung sollte aufgrund der Wartungsfreundlichkeit auch für einen längeren Zeitraum ausgelegt werden. Durch die hierfür notwendige Kapazität ist es nahezu unmöglich, diese sehr klein und leicht zu gestalten, um die Messung nicht zu sehr zu beeinflussen. Die zweite Methode mit einer externen Stromversorgung würde logischerweise eine Verbindung über ein Kabel zwischen dem sich bewegenden Gleiter und einem fest

montierten Netzgerät mit sich bringen. Da dieses einerseits die Messung stark beeinflussen kann und der Aufbau außerdem weder robust noch kompakt mit einem freischwingen Kabel umgesetzt werden kann, ist ein Beschleunigungssensor für diesen Anforderungsbereich nicht optimal geeignet.

2.1.2 Magnetostriktive Sensoren

Magnetostriktive Sensoren arbeiten mit der Kopplung von Magnetfeldern. Es wird durch einen Metallstab, auf dem ein beweglicher Permanentmagnet befestigt ist, an einem Ende ein Stromimpuls hineingeschickt. Der Impuls wiederum erzeugt ebenfalls ein Magnetfeld. Sobald sich dieses mit dem Feld des Magneten überlagert, entsteht in der Kristallstruktur des Stabes eine elastische Verformung, welche sich in Form einer Welle ausbreitet. Durch die Zeitdifferenz vom Aussenden des Stromimpulses und der Ankunft der mechanischen Welle am Ende des Stabes lässt sich anschließend die aktuelle Position des Permanentmagnet berechnen. Würde man diesen Magneten nun auf dem Gleiter befestigen und den Stab an der Luftkissenbahn, würde sich dies hervorragend zur Datenerfassung für die Schwingung eignen. Das Problem ist bei dieser Variante jedoch die Wirbelstrombremse. Diese arbeitet, wie bereits unter 2.1.4 erklärt ebenfalls mit Magnetfeldern und ist direkt an der Luftkissenbahn positioniert. Aus diesem Grund ist es sehr wahrscheinlich, dass durch den Betrieb der Bremse die Überlagerung der Magnetfelder verändert wird und dadurch die Daten verfälscht werden. Deswegen wurde dieser Sensortyp ebenfalls verworfen.[2]

2.1.3 Magnetbandsensoren

Ein Magnetbandsensor besteht grundsätzlich aus zwei Teilen. Einem flexiblen Band mit einer Magnetträgerschicht und einem Sensorkopf. Das Band ist in regelmäßigen Abständen abwechselnd mit magnetischen Nord- und Südpolen magnetisiert. Der Sensorkopf gleitet auf diesem Band und erfasst anhand der Anzahl der durchlaufenen Nord- und Südpolen die Entfernung, die er zurückgelegt hat. Der Nachteil hierbei ist, dass der bewegliche Sensorkopf auf dem Gleiter ebenfalls mit einem Kabel verbunden ist. Das frei schwingende Kabel würde wiederum die Robustheit des Aufbaus einschränken und außerdem auch die Messung beeinflussen. Darum wurde auch dieser Sensortyp aus der Auswahl genommen.[3]

2.1.4 Ultraschallsensoren

Ultraschallsensoren können durch die Umrechnung der Laufzeit des Schalls zur Entfernungsmessung verwendet werden. Hierbei wird vom Sensor ein Signal ausgesendet und wird von einem Hindernis oder Gegenstand zum Sensor zurück reflektiert. Durch die Zeitdifferenz und die bekannte Ausbreitungsgeschwindigkeit kann daraus die Entfernung bestimmt werden. Der Vorteil dieser Sensorart liegt darin, dass der Sensor

nicht selbst auf dem Gleiter positioniert werden muss. Es reicht auf dem Gleiter einen kleinen Detektorschirm zu befestigen und den Sensor am Ende der Luftkissenbahn zu montieren. Dies ermöglicht einen robusten und kompakten Aufbau sowie eine stabile Stromversorgung. Deshalb wurde dieser Sensortyp im Anschluss noch genauer untersucht.[4]

2.1.5 Optische Sensoren

Optische Sensoren arbeiten mit einem ähnlichem Prinzip wie Ultraschallsensoren. Der Sensor sendet einen Impuls, dieser wird an einem Gegenstand reflektiert und ein Teil davon trifft wieder auf den Sensor selbst auf. Daraus lässt sich im Nachgang die Entfernung des Gegenstandes, an dem das Licht reflektiert wurde, bestimmen. Das heißt, auch hier wäre es möglich den Sensor am Ende der Luftkissenbahn zu montieren und auf dem Gleiter selbst nur einen kleinen Schirm zu befestigen.

2.1.6 Fazit

Die Abwägung der Vor- und Nachteile führte zur Entscheidung, einen Ultraschallsensor und einen optischen Sensor noch genauer auf ihre Tauglichkeit zu untersuchen.

2.2 Auswahl des Sensors

Bei diesen Tests wurde der Schwerpunkt auf die Messfrequenz, die Messgenauigkeit und die Reproduzierbarkeit der einzelnen Messwerte gelegt.

Getestet wurden zuerst zwei verschiedene Sensoren, beide sind relativ günstige Einsteigermodelle, wobei es sich um einen Ultraschallsensor und einen optischer Sensor handelt. Der 'HC-SR04' ist ein Ultraschallsensor zur Entfernungsmessung und der optische Sensor 'GP2Y0A21YK0F' der Firma Sharp ist ein Infrarotsensor, der Entferungen mithilfe der Triangulationsmethode bestimmt.

2.2.1 Ultraschallsensor HC-SR04

2.2.1.1 Funktionsweise und Spezifikationen

Der Sensor HC-SR04 ist ein Ultraschallsensor, welcher eine Versorgungsspannung von $4.5 - 5.5V$ benötigt. Er kann Entfernungen in einem Bereich von etwa $2 - 300cm$ erfassen und hat hierbei eine maximale Abweichung von circa $3mm$. Der Sensor besitzt einen Sender (S) und einen Empfänger (E), wie in folgender Grafik zu sehen ist.

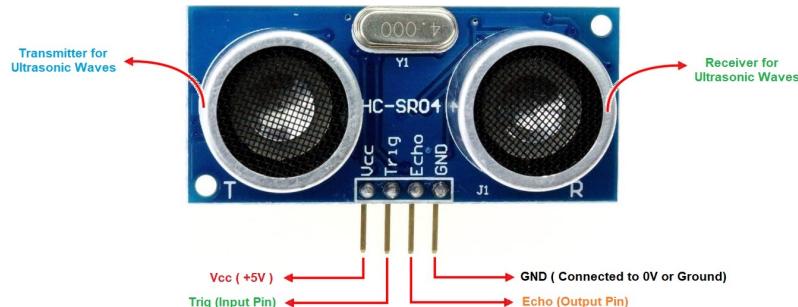


Abbildung 2.1: Aufbau des Ultraschallsensors[5]

Neben der Spannungsversorgung existieren noch zwei weitere Anschlüsse, der Trigger(Input) und das Echo(Output). Um eine Messung durchzuführen, muss das Signal am Triggereingang für mindestens $10\mu s$ abfallen. Nach $250\mu s$ wird daraufhin ein kurzer Ultraschallimpuls ausgesendet. Dieser wird dann an einem Gegenstand reflektiert und ein Teil davon gelangt dabei auch zum Empfänger, wie in folgender Abbildung gezeigt:

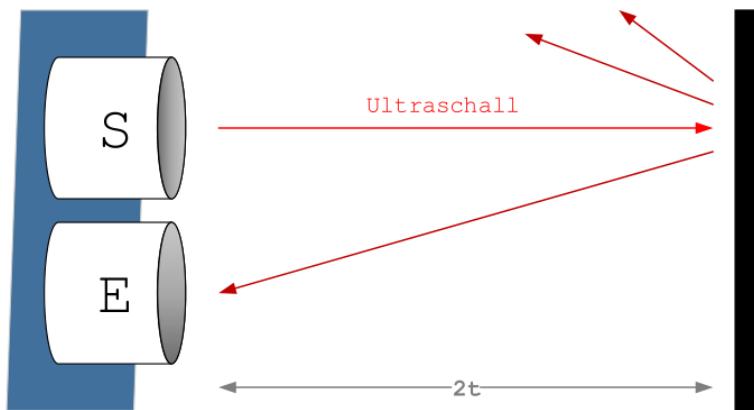


Abbildung 2.2: Schematische Funktionsweise eines Ultraschallsensors[6]

Am Echo-Ausgang liegt eine Spannung an, sobald das Ultraschallsignal ausgesendet wird. Die Spannung fällt wieder ab, sobald das Signal am Empfänger ankommt. Dieser Vorgang ist mithilfe eines Oszilloskopes sehr gut zu erkennen:

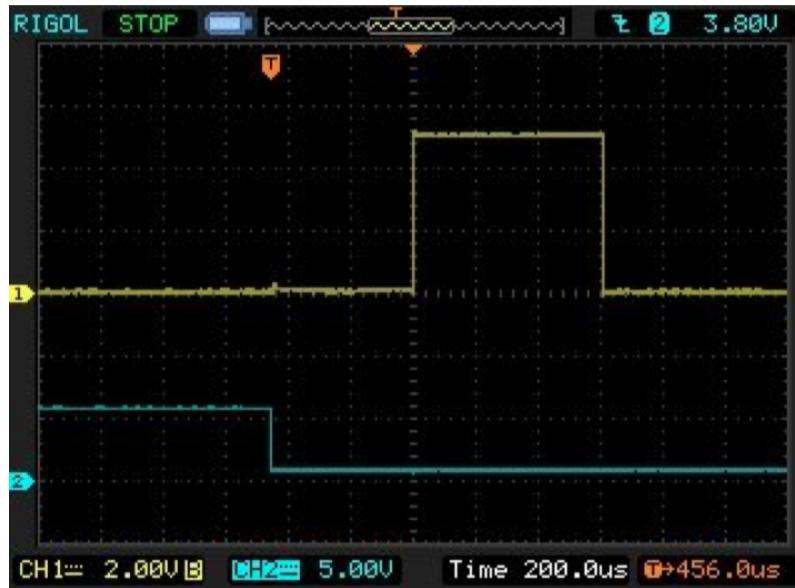


Abbildung 2.3: Ein- und Ausgangssignal des HC-SR04[7]

Kanal 1(gelb) ist hierbei der Trigger und Kanal 2(blau) das Echo, also der Ausgang des Sensors. Sobald das Trigger-Signal kurz abfällt, wird der Echo-Ausgang $250\mu s$ später auf HIGH geschalten. In diesem Beispiel war der Schall $t_{schall} = 600\mu s$ unterwegs, bis er am Empfänger wieder ankam. Am Ausgang lag somit $600\mu s$ eine Spannung an. Mit der bekannten Schallgeschwindigkeit v_{schall} von $343 \frac{m}{s}$ lässt sich daraus die zurückgelegte Entfernung bestimmen.

$$d_2 = v_{schall} \cdot t_{schall} = 20.6cm \quad (2.1)$$

Da der Schall die Entfernung zum Objekt zweimal zurückgelegt hat (Hin- und Rückweg), muss dieser Wert nun noch halbiert werden:

$$d = \frac{d_2}{2} = 10.3cm \quad (2.2)$$

Somit beträgt die Entfernung zum Objekt in diesem Beispiel $10.3cm$

Sollte der Empfänger kein Signal erhalten, so wartet der Sensor $200ms$ und ist danach wieder bereit für eine neue Messung. Da in unserem Versuchsaufbau der Detektionsschirm fest montiert ist und im rechten Winkel zu der Ausbreitungsrichtung der Wellen steht, dürfte dieser Fall nicht vorkommen. Der maximale Abstand zwischen Sensor und Schirm ist ausserdem niemals größer als $1m$, daraus folgt für die

maximale Schalllaufzeit:

$$t_{max} \frac{d}{v_{schall}} = \frac{1m}{343\frac{m}{s}} = 2.9ms \quad (2.3)$$

Dies bedeutet, auch durch die maximale Schalllaufzeit ist es nicht möglich, dass der Sensor $200ms$ warten muss. Möchte man nun die Messung kontinuierlich durchführen, legt man an den Trigger ein rechteckiges Signal mit einer Frequenz von größer gleich $50Hz$ an. Diese Frequenz kann nicht unterschritten werden, da der Sensor nur alle $20ms$ eine Messung durchführen kann.

[7]

2.2.1.2 Anschluss am Raspberry Pi

Zur Datenauswertung wird der Sensor am Raspberry Pi angeschlossen. Hierfür werden zwei GPIO Pins benötigt. Diese 'General Purpose Input/Output' Anschlüsse sind programmierbare Kontaktstifte, welche als Schnittstelle zwischen dem Raspberry Pi und anderen Systemen genutzt werden können. Mit dem einen wird das Triggersignal erzeugt und mit dem anderen das Echo ausgelesen. Da nicht die genaue Ausgangsspannung des Sensors wichtig ist, sondern nur die HIGH bzw LOW Stellung, also ob Spannung anliegt oder nicht, benötigt man auch keinen Analog Digital Wandler in der Schaltung. Diese Schaltung im Folgenden schematisch dargestellt:

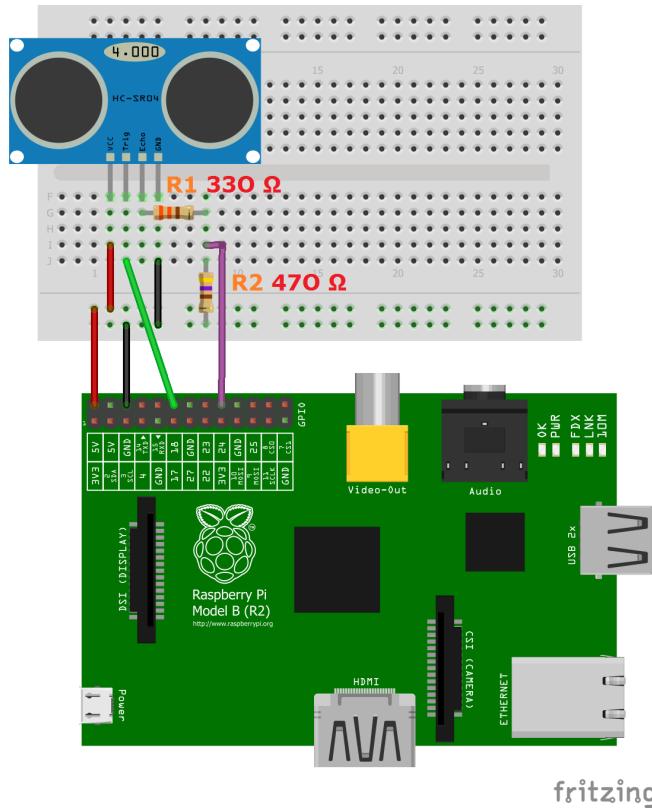


Abbildung 2.4: Schaltplan des Ultraschallsensors HC-SR04 am Raspberry Pi[7]

Die beiden Widerstände fungieren dabei als Spannungsteiler. Die Funktionsweise von diesen wird im Anschluss noch kurz erläutert. Der Input am Raspberry würde theoretisch bis zu 3.3V vertragen, durch den Spannungsteiler werden die 5V des Sensors auf 2.9V reduziert, diese reichen jedoch auch aus, sodass der Input auf HIGH geschalten wird. Der Code zur Ansteuerung des Sensors befindet sich im Anhang.

Spannungsteiler Hat man ganz allgemein an einer Apparatur eine Spannung U_{ges} anliegen, welche für die weitere Verarbeitung viel zu groß ist, dient oftmals ein Spannungsteiler als guter Lösungsansatz. Beispielsweise erlauben die GPIO des Raspberry Pi's nur maximal 3.3V. Der Sensor HC-SR04 selbst läuft aber mit 5V. Somit wäre es möglich, dass der Ausgang des Sensors den Raspberry zerstört. Hier teilt man nun die Spannung auf, das heißt, man schließt zwei Widerstände an und greift über einen der beiden dann die Spannung ab.

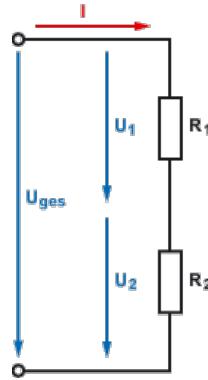


Abbildung 2.5: Aufbau Spannungsteiler[8]

Nach dem Ohmschen Gesetz gilt folgender Zusammenhang:

$$U = R \cdot I \quad (2.4)$$

Außerdem addieren sich Widerstände in einer Reihenschaltung, somit folgt:

$$R_{ges} = R_1 + R_2 \quad (2.5)$$

$$I_{ges} = \frac{U_{ges}}{R_{ges}} = I_1 = I_2 \quad (2.6)$$

Somit ergibt sich für die Spannung an R_2 :

$$U_2 = R_2 \cdot I_2 = R_2 \cdot \frac{U_{ges}}{R_{ges}} \quad (2.7)$$

und durch Umformung erhält man:

$$\frac{U_2}{U} = \frac{R_2}{R_{ges}} \quad (2.8)$$

Hiermit lässt sich durch geeignete Wahl der Widerstände R_1 und R_2 die maximale Spannung, die letztendlich über den Widerstand R_2 ausgelesen wird, reduzieren.

Betrachtet man nun die beiden Widerstände $R_1 = 330\Omega$ und $R_2 = 470\Omega$ aus der Schaltung, so ergibt sich:

$$\frac{U_2}{U} = \frac{R_2}{R_{ges}} = 0.5875 \quad (2.9)$$

Geht man daraufhin von einer maximalen Spannung U_{ges} von 5V aus, folgt daraus:

$$U_2 = 0.5875 \cdot U_{ges} = 2.9V \quad (2.10)$$

2.2.1.3 Messwiederholungsrate

Wie bereits unter 2.2.1.1 erwähnt, kann der Sensor alle $20ms$ einen Wert liefern, somit wäre die Messfrequenz bei $50Hz$. Dieser Wert übertrifft bereits deutlich die aktuelle Messfrequenz des Funkenschreibers, somit ist der Sensor schnell genug um keine qualitativen Einbußen hinnehmen zu müssen.

2.2.1.4 Fokussierung des Messbereiches

Nachdem die Messfrequenz des Sensors völlig ausreichend ist, wurde im nächsten Schritt der Fokus des Sensors getestet. Laut Herstellerangaben öffnet sich der Schallkegel mit einem Winkel von 15° öffnen. Hierbei wurde der Sensor mithilfe eines 3D-gedrucktem Gehäuses auf einer Holzplatte befestigt. Diese Platte war lackiert, sodass Reflexionen von der Oberfläche zurück zum Sensor vermieden werden. Außerdem wurden in regelmäßigen Abständen von $10cm$ Markierungen auf der Platte angebracht, um das Messen zu erleichtern:

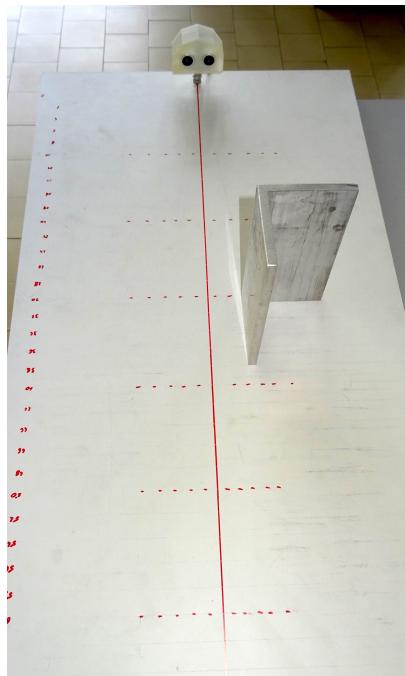


Abbildung 2.6: Aufbau zur Bestimmung der Fokussierung des Messbereiches

Nun wurde bei verschiedenen axialen Abständen jeweils von beiden Seiten der Aluminiumwinkel in die Mitte gefahren und die radiale Distanz zur Mitte vermerkt,

sobald der Sensor den Winkel detektiert hatte. Die Schwierigkeit hierbei war, den Winkel immer senkrecht zur axialen Ausbreitungsrichtung zu halten. Minimale Abweichungen ergaben bereits Messunterschiede von mehreren Zentimetern. Dabei ergeben sich folgende Messwerte:

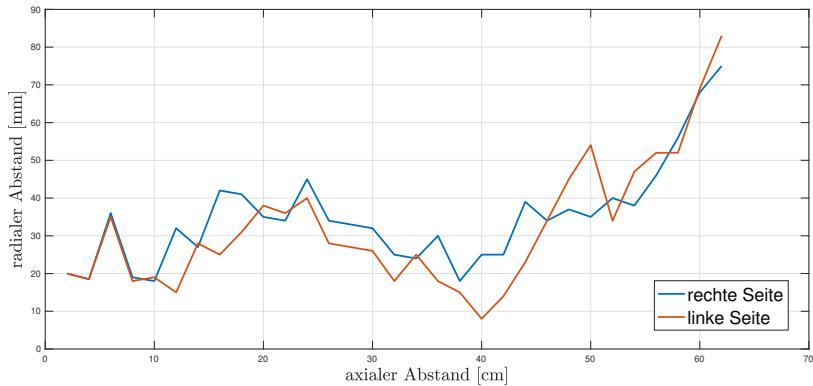


Abbildung 2.7: Messdaten zum Test der Fokussierung des Messbereiches

Theoretisch müssten beide Graphen übereinstimmen und linear ansteigen. Dies ist auf Basis der Messergebnisse nicht zu erkennen. Die Fehler lassen sich jedoch durch die schwierige manuelle Justierung des Winkels zwischen der Oberfläche des Messobjektes und dem Sensor erklären. Sobald der Winkel nur minimal verändert wird, lassen sich aufgrund der veränderten Reflexion auch ganz andere Messbereiche bestimmen.

Ein weiterer Punkt ist, dass selbst bei viel größeren radialen Abständen unter einem bestimmtem Winkel immer eine Fehlmessung stattfindet. Dies ist der Fall, wenn die Oberfläche des Messobjektes genau senkrecht zu der Verbindungsachse zum Sensor steht. Dies trifft tritt selbst außerhalb des angegebenen Schallkegels von 15° auf und ist in folgender Abbildung grafisch dargestellt:

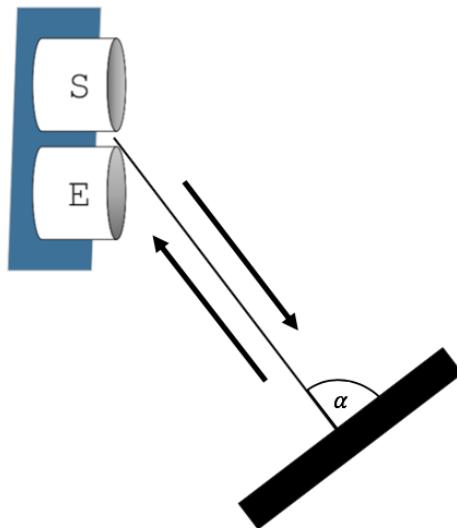


Abbildung 2.8: Reflexion des Ultraschallimpulses

Für $\alpha = 90^\circ$ wird die Reflexion in Richtung des Sensors maximal. So war es beispielsweise möglich, bei einem axialen Abstand von 20cm den Aluminiumwinkel bereits bei einem radialen Abstand von 14cm zu erfassen. Dies entspricht einem Öffnungswinkel β von 35° :

$$\beta = \arctan \left(\frac{14\text{cm}}{20\text{cm}} \right) = 35.0^\circ \quad (2.11)$$

Dies bedeutet, dass der Sensor in seiner jetzigen Form definitiv nicht für den Versuch geeignet ist, da die äußeren Einflüsse viel zu leicht Fehlermessungen verursachen können. Aus diesem Grund wurde daraufhin versucht, den Öffnungswinkel unter Verwendung von seitlichen Barrieren zu verkleinern. Deswegen wurden zwei weitere Winkel angefertigt, welche an der Innenseite mit Malerflies verkleidet waren, um wiederum Reflexion zu vermeiden:

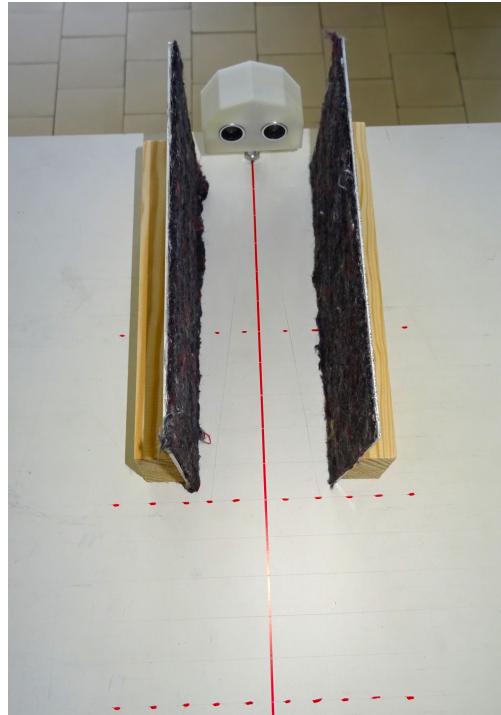


Abbildung 2.9: Ultraschallsensor mit Barrieren

Der Einsatz dieser Fokussierung hat sehr gut funktioniert und es waren außerhalb des Öffnungswinkels der Barrieren fast keine Messungen mehr möglich. Der Aluminiumwinkel wurde bei kleinerer axialer Entfernung nur innerhalb dieses Winkels und bei größerer axialer Entfernung nur minimal daneben erfasst. Somit lässt sich abschließend feststellen, dass der Ultraschallsensor ohne eine zusätzliche Fokussierung nicht geeignet ist, mit einer externen Barriere jedoch gut funktioniert.

2.2.1.5 Messgenauigkeit/Reproduzierbarkeit

Abschließend wurde noch die Reproduzierbarkeit der Messwerte getestet und die mit 3mm angegebene Messgenauigkeit überprüft. Hierfür wurde das Gehäuse mit dem Sensor auf einer optischen Bank befestigt und der Abstand zu einem verschiebbaren Spiegel bestimmt:

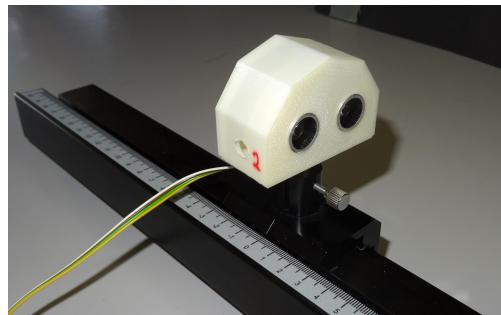


Abbildung 2.10: Befestigung des Ultraschallsensors auf der optischen Bank

Der Abstand wurde in einem Intervall von $6\text{--}64\text{cm}$ mit 2 verschiedenen Messreihen bestimmt. Die Ergebnisse sind in folgendem Plot zu sehen:

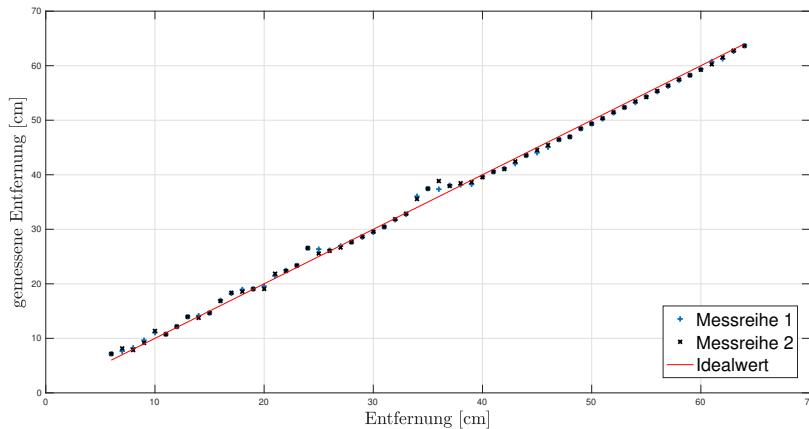


Abbildung 2.11: Messwerte zur Messgenauigkeit des Ultraschallsensors

Die rote, durchgezogene Linie entspricht dem Idealwert. Man kann den linearen Anstieg bei beiden Messreihen sehr gut erkennen, auch wenn beide Reihen immer wieder größere Abweichungen enthalten. Der Mittelwert der Abweichungen zwischen den beiden Messreihen beträgt circa 6.9mm . Selbst wenn man eine Messgenauigkeit durch das Ablesen der Distanz auf der optischen Bank von einem Millimeter annehmen würde, ist diese Abweichung immer noch zu groß.

Ein weitere Ungenauigkeit ist ganz allgemein beim Verschieben des Spiegels festzustellen. Wird mit diesem mehrmals eine bestimmte Postion angefahren, dazwischen der Spiegel jedoch kurz aus dem Messbereich des Sensors genommen, ergeben sich unterschiedliche Messwerte für gleiche Spiegelpositionen.

2.2.1.6 Fazit

Abschließend lässt sich feststellen, dass der Sensor Typ Ultraschallsensor mit einer zusätzlichen Fokussierung sehr gut für diesen Anwendungsbereich geeignet ist. Das untersuchte Modell HC-SR04 kann hierfür jedoch nicht verwendet werden, da es eine viel zu große Messgenauigkeit aufweist.

2.2.2 IR-Sensor Sharp GP2Y0A21YK0F

2.2.2.1 Funktionsweise und Spezifikationen

Der Infrarotsensor der Firma Sharp arbeitet mit der Triangulationsmethode. Diese bestehen aus einer Infrarot-LED, einer Photodiode und zwei Sammellinsen. Die LED emittiert Licht im Infrarotbereich und dieses wird durch die erste Sammellinse kolliamt. Dies bedeutet, die Strahlen verlaufen nun senkrecht zueinander. Daraufhin wird das Licht an einem Gegenstand reflektiert und ein Teil der Strahlung erreicht dabei auch wieder die Photodiode. Vor der Photodiode werden die Strahlen durch eine zweite Sammellinse noch einmal fokussiert:

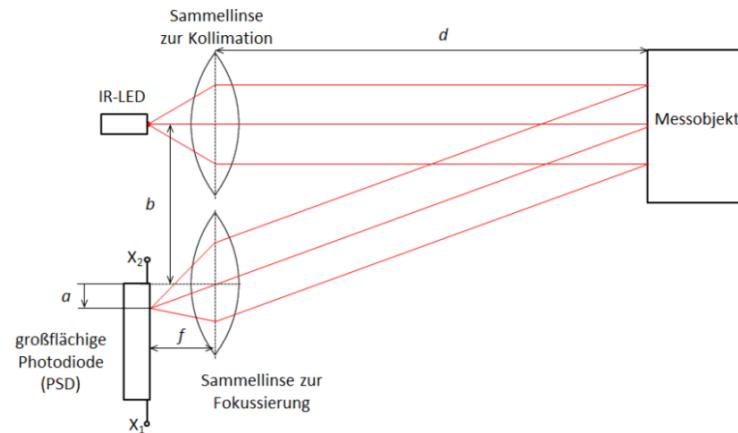


Abbildung 2.12: Strahlenverlauf bei der Infrarot-Triangulation[9]

Durch Anwendung des Strahlensatzes lässt sich nun die Entfernung zu dem Gegenstand, an dem das Licht reflektiert wurde, bestimmen.

$$\frac{f}{a} = \frac{d}{b} \quad (2.12)$$

$$\Rightarrow d = \frac{f \cdot b}{a} \quad (2.13)$$

Die Variablen b und f lassen sich hierbei relativ leicht bestimmen. Die Brennweite ist bekannt und der Abstand von LED zur Diode lässt sich genau messen bzw. ist bereits durch die Auslegung und Herstellerangabe bekannt. Das Problem ist die Variable a . Hier macht man sich nun zu eigen, dass die Diode keine normale Photodiode ist, sondern eine großflächige Position-Sensitive-Device Diode. Diese hat eine große lichtsensitive Fläche mit einheitlichem Flächenwiderstand.

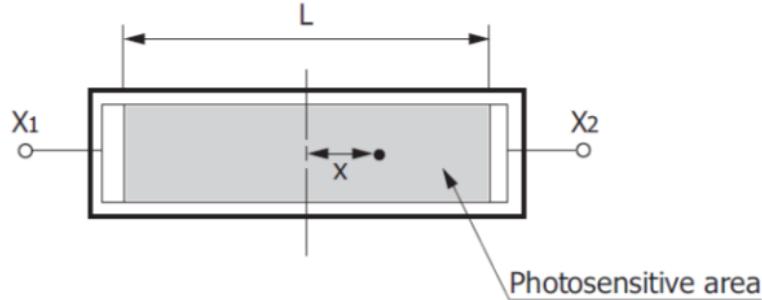


Abbildung 2.13: Skizze der Photodiode[9]

Trifft nun auf eine Position x ein Lichtstrahl auf, so entstehen 2 Ströme I_1 und I_2 , die an den Kontakten X_1 und X_2 abfließen. Da der Flächenwiderstand konstant ist, gilt folgende Beziehung zwischen den Strömen und der Länge der Diode L sowie der Variablen x :

$$\frac{x}{\frac{L}{2}} = \frac{I_2 - I_1}{I_2 + I_1} \quad (2.14)$$

Drückt man die Variable x nun noch in Abhängigkeit der Parameter L und a aus, folgt daraus:

$$x = \frac{L}{2} - a, \left(0 \leq a \leq \frac{L}{2} \right) \quad (2.15)$$

Dies ergibt dann:

$$\frac{\frac{L}{2} - a}{\frac{L}{2}} = \frac{I_2 - I_1}{I_2 + I_1} \quad (2.16)$$

$$\Rightarrow a = L \cdot \frac{I_1}{I_1 + I_2} \quad (2.17)$$

Da nun alle drei Parameter a , b und f bekannt sind, lässt sich daraus die Entfernung zum Messobjekt bestimmen:

$$d = \frac{f \cdot b}{a} = \frac{f \cdot b}{L} \cdot \frac{I_1 + I_2}{I_1} \quad (2.18)$$

Der Sensor gibt dann im Anschluss ein analoges Signal aus, welches im direkten Zusammenhang mit der gemessenen Entfernung steht. Folgende Messung wurde bereits vom Hersteller durchgeführt und zeigt die Ausgangsspannung in Abhängigkeit der Entfernung bei einem weißen und einem grauen Papier mit unterschiedlichen Reflexionsindices:

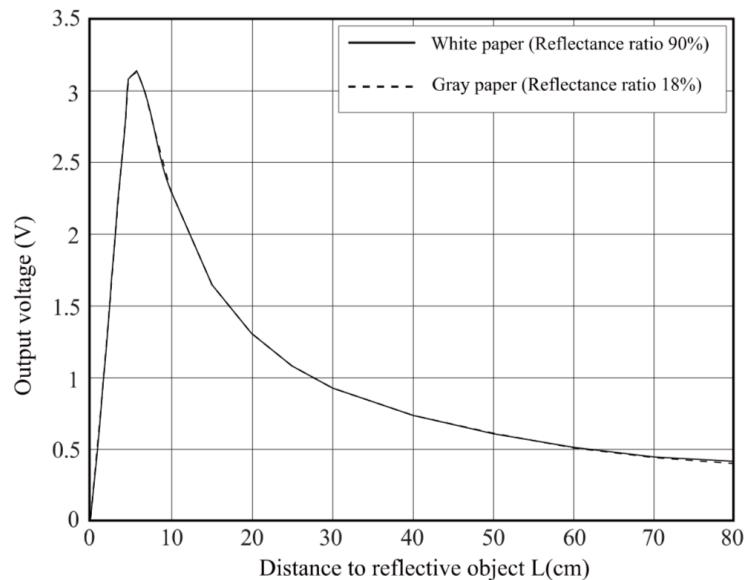


Abbildung 2.14: Ausgangskennlinie des Infrarotsensors für weißes und graues Papier[10]

Die Besonderheit dieser Sensorart lässt sich an dem Graphen bereits sehr gut erkennen. Auch wenn dies ein optischer Sensor ist und die Reflexion des Lichtes immer vom Material des Messobjektes abhängig ist, hat dies keinerlei Auswirkung auf die Ausgangsspannung, solange an dem Material Licht reflektiert wird. Da diese Spannung nur von der Position des Lichtstrahls und nicht der Intensität abhängt, liefert der Sensor gleiche Ausgangswerte auch für unterschiedliche Oberflächen. Der Nachteil ist hierbei, dass bereits minimale Drehungen des Messobjektes eine sehr große Veränderung auf die Ausgangsspannung haben können. Prinzipiell ist dies für die

Anwendung nicht von Bedeutung, solange der Winkel konstant ist. Denn der Sensor besitzt nur einen analogen Ausgang und dieser muss nachher noch in die Entfernung umgerechnet werden. Bei der Umrechnung kann dieser Umstand im Nachgang durch einen Korrekturfaktor berücksichtigt werden. Problematisch kann es nur werden, wenn sich das Messobjektes während einer Messung etwas dreht. In diesem Fall ist es nicht eindeutig bestimmbar, ob die Änderung der Ausgangsspannung auf eine Distanzänderung oder eine Drehung des Messobjektes zurückzuführen ist. [9] [10]

2.2.2.2 Anschluss am Raspberry Pi

Zur Datenauswertung wurde auch dieser Sensor an den Raspberry Pi angeschlossen. Da dieser jedoch keine analogen Eingänge hat, war hier noch ein Analog-Digital-Converter, auch ADC oder Analog-Digital-Wandler genannt, notwendig. Da in Weihenstephan bereits einige Analog-Digital-Wandler des Typs 'MCP3008' vorhanden waren, wurden diese hierfür auch verwendet. Der MCP3008 teilt die Eingangsspannung (in diesem Fall $3.3V$) in 1024 Bereiche auf. Der kleinste Wert ist hierbei 0 und der größte 1023. Diese können anschließend am Raspberry ausgegeben werden. Natürlich lassen sich die 1024 Bereiche, nun B genannt, auch wieder in die Spannung umrechnen:

$$U_{out} = \frac{B}{1023} \cdot 3.3V \quad (2.19)$$

Die Spannung, beziehungsweise diese 1024 Bereiche, können anschließend mithilfe der Abbildung 2.14 auch noch in die Entfernung umgerechnet werden. Dieser IC wird mit dem SPI Bus des Raspberry angesteuert. Die Schaltung sieht hierbei folgendermaßen aus:

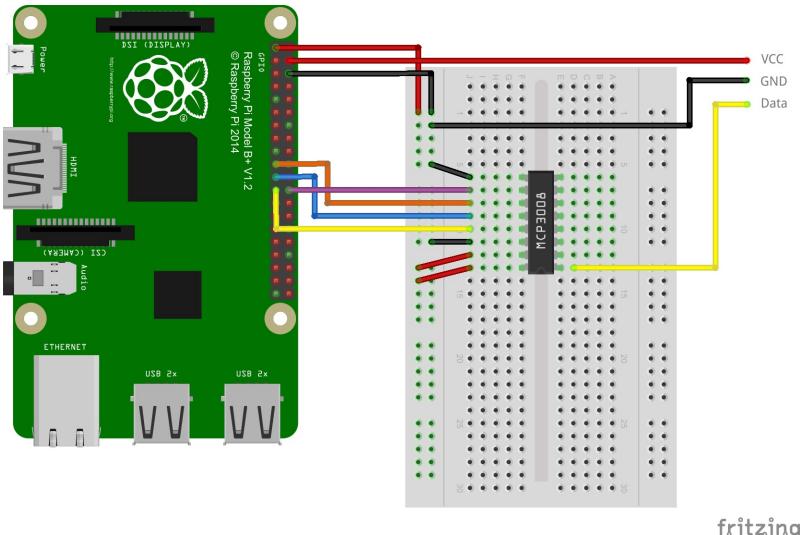


Abbildung 2.15: Schaltplan des Infrarotsensors am Raspberry Pi[11]

Auf der linken Seite befindet sich die Steuerung für den SPI Bus. Auf der rechten Seite besitzt der integrierte Schaltkreis acht analoge Eingänge, wobei hier nur ein einziger verwendet wird. Der Code zur Ansteuerung befindet sich ebenfalls im Anhang.

2.2.2.3 Messwiederholungsrate

Der Infrarotsensor benötigt für eine Messung $38.3 \pm 9.6ms$. Die Ausgangsspannung wird daraufhin nach maximal $5ms$ aktualisiert. Die Verzögerung von circa $5ms$ ist für diese Anwendung jedoch nicht weiter von Bedeutung, da die Daten erst im Anschluss an die Messung ausgewertet werden. Unterstellt man eine mittlere Messdauer von $38.3ms$, ergibt sich daraus, dass der Sensor mit einer Frequenz von $26.1Hz$ messen kann. Dies ist etwas langsamer als die bisherige Messfrequenz. Wenn man jedoch berücksichtigt, dass aktuell selbst bei einer sehr schnellen Schwingung noch sehr viele Messpunkte vorhanden sind, sollte diese Messfrequenz immer noch ausreichend sein, um die Schwingungskurve im Anschluss darstellen zu können.

2.2.2.4 Messgenauigkeit/Reproduzierbarkeit

Um die Abbildung 3.13 gezeigte Abhängigkeit von Ausgangsspannung und Entfernung zu testen, wurde der Infrarotsensor ebenfalls mithilfe einer Halterung und einem Schlitten auf einer optischen Bank montiert. Auf einem zweiten Schlitten wurde ein Spiegel befestigt, um die Entfernung zwischen diesem und dem Sensor zu messen:



Abbildung 2.16: Montage des Infrarotsensor auf der optischen Bank

Daraufhin wurde die Spannung in Abhängigkeit der Entfernung für einen Bereich von 3.5cm bis 60cm gemessen. Die Ergebnisse sind in folgendem Plot zu sehen:

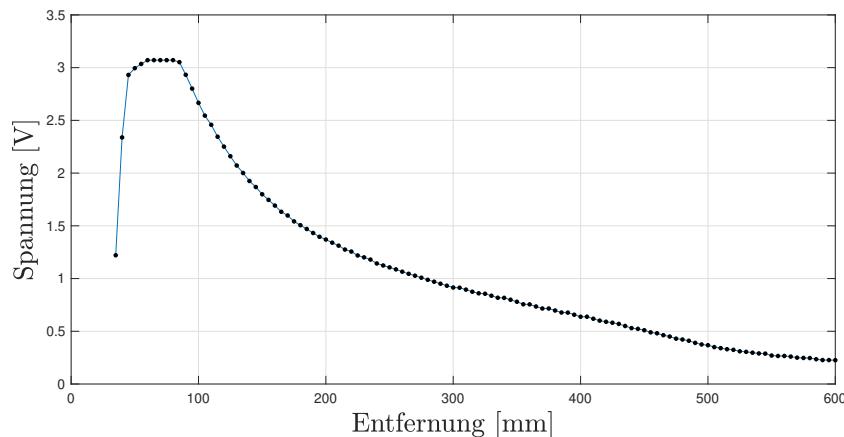


Abbildung 2.17: Ausgangsspannung des Infrarotsensors in Abhängigkeit der Entfernung

Sowohl der Anstieg der Spannung für kleine Entfernungen als auch der langsame Abfall nach einem Maxima sind hierbei sehr gut zu erkennen. Auch die Tatsache, dass eine sinnvolle Abstandsmessung ab circa 10cm möglich ist, wird an der grafischen Darstellung ersichtlich. Denn erst bei Eingrenzung des Entfernungsintervalls auf Distanzen größer als $\approx 10\text{cm}$ ist die eindeutige Zuordnung der Entfernung bei einer bestimmten Spannung gegeben.

Um nun die Reproduzierbarkeit der Ergebnisse zu testen, wurde diese Messung auf einem Intervall von $30 - 60\text{cm}$ wiederholt. Dieses Intervall wurde so festgelegt, da dies dem Bereich entspricht, der für den Anwendungsbereich von Interesse ist. Das Resultat ist in folgender Grafik zu sehen:

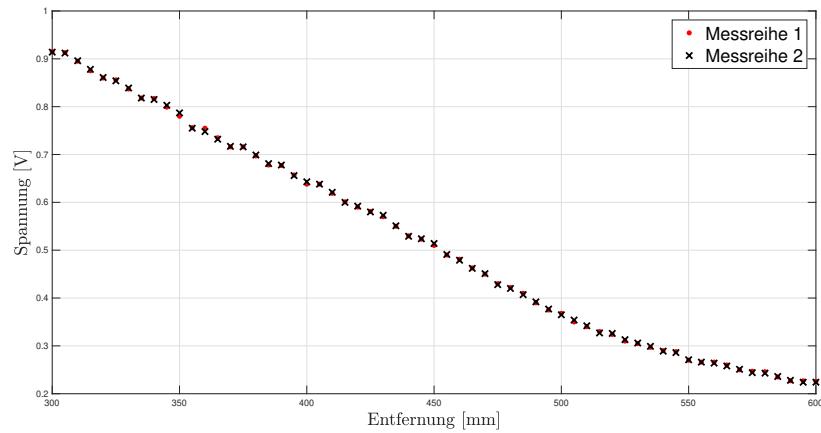


Abbildung 2.18: Ausgangsspannung des Infrarotsensors in Abhängigkeit der Entfernung bei zwei Messreihen

Wie man erkennen kann, liegen die Messwerte fast immer nahezu exakt aufeinander. Die Schwierigkeit beim Bestimmen der Ausgangsspannung war, dass die Messwerte am Spannungsmessgerät meist etwas geschwankt haben, hier wurde dann versucht einen sinnvollen Mittelwert zu finden. Die Abweichung der Ausgangsspannung bei beiden Messreihen liegt bei maximal 1.3%. Ergänzend ist anzumerken, dass diese Ergebnisse nur durch eine Fixierung des Spiegels auf der optischen Bank möglich waren. Der Spiegel war auf einem Schlitten befestigt, welcher auf der optischen Bank gleiten kann. Damit sich der Schlitten bewegen kann, muss dieser etwas locker auf der Bahn aufgesetzt werden. Dadurch war auf der Bahn eine minimale Drehung des Spiegels möglich. Dies reichte bereits aus, um die Ausgangsspannung zu beeinflussen. Somit wurde der Spiegel immer in Position gebracht und daraufhin mit der Feststellschraube an der Bank fixiert. Hierdurch konnte eine seitliche Drehung des Spiegels verhindert werden.

Außerdem ist an der Kurve sehr gut zu erkennen, dass bei größeren Entfernungen die Spannungsunterschiede immer geringer werden. Da wir eine Auflösung von mindestens 1mm anstreben wollen, wird dies bei größeren Entfernungen ebenfalls zum Problem. Hierbei entstand nun die Idee, den Spiegel von Grund auf etwas

schräg zu montieren, da das Maximum der Spannung bei einer bestimmten Distanz nicht bei einem rechten Winkel zustande gekommen ist. Das bedeutet, bei einer leicht schrägen Montage können höhere Spannungswerte und damit auch eine höhere Auflösung erreicht werden. Um den Einfluss der seitlichen Drehung noch genauer zu untersuchen, wurde anschließend noch eine weitere Messung durchgeführt. Hierbei wurde der Spiegel wiederum auf einem Schlitten montiert, nun jedoch beweglich. Das heißt, der Spiegel lies sich nun seitlich drehen und der Auslenkungswinkel war an Hand eines Zeigers sowie einer Grad-Schablone ablesbar:

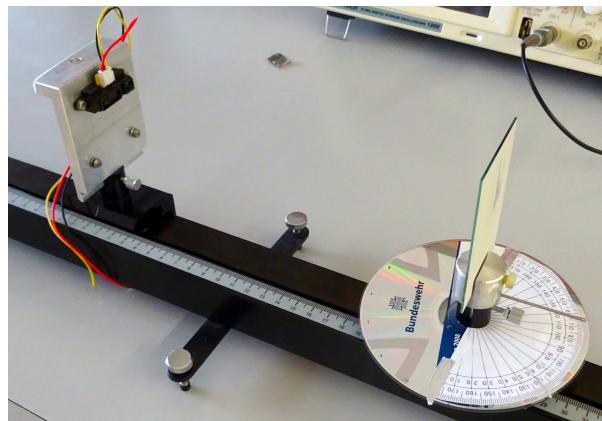


Abbildung 2.19: Aufbau zur seitlichen Drehung des Spiegels

Für verschiedene Abstände wurden hierbei die Winkel erfasst, bei denen die Ausgangsspannung ihr Maximum hatte:

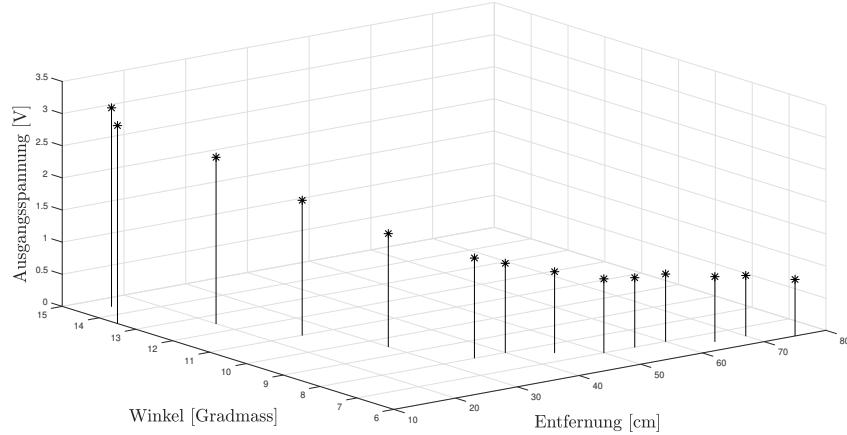


Abbildung 2.20: Winkel der maximalen Ausgangsspannung und maximale Ausgangsspannung in Abhängigkeit der Entfernung bei dem Infrarotsensor

Man erkennt, dass sich die maximale Ausgangsspannung mit zunehmender Entfernung verringert. Vergleicht man diese Werte nun mit der Kurve aus Abbildung 2.18 lässt sich sehr deutlich erkennen, dass die maximale Spannung jedoch teilweise sogar noch verdreifacht werden kann (vgl. Distanz 45cm mit 0.5V und 1.5V). Das Problem hierbei ist jedoch, dass die Spannungsänderung bei diesen Maxima bei einer minimalen Drehung noch viel stärker ausfällt als bei einer Montage des Spiegels in einem rechten Winkel.

2.2.2.5 Fazit

Somit lässt sich zusammenfassend feststellen, dass der Sensor eine gute Reproduzierbarkeit besitzt, da bei zwei aufeinander folgenden Messreihen die maximale Abweichung niemals mehr als 1.3% betrug. Dies gilt aber nur, solange der Schirm absolut keine seitliche Ablenkung erfährt. Die Auflösung selbst lässt sich durch eine konstante Drehung des Schirmes erhöhen, dies erhöht jedoch auch die Messfehler aufgrund einer minimalen Veränderung des Winkels von Sensor zu Schirm. Da die Messung selbst auf einer Luftpumpe erfolgt und der gesamte Aufbau der individuellen Bedienung verschiedener Studenten ausgesetzt ist, kann nicht ausgeschlossen werden, dass durch unbeabsichtigte Stöße oder unvorsichtiges Bedienen eine minimale Verschiebung des Aufbaus erfolgt. Aus diesem Grund ist auch dieser Sensor nicht für diesen Anwendungsbereich geeignet.

2.2.3 Ultraschallsensor UFA-1500-M18-A

2.2.3.1 Funktionsweise und Spezifikationen

Der zweite Ultraschallsensor ist ein Sensor zur Abstandsmessung der Firma WayCon aus der UFA-1500 Serie. Diese haben einen Messbereich von 120mm bis 1500mm , welcher zusätzlich noch kalibrierbar ist. Die Auflösung dieser Serie beträgt 0.5mm und der hier verwendete Sensor besitzt eine axiale Messrichtung mit einem Analogausgang von $0 - 10\text{V}$ oder $4 - 20\text{mA}$. [12]

2.2.3.2 Anschluss am Raspberry Pi

Da der Sensor genau wie der bereits untersuchte Infrarotsensor einen analogen Ausgang besitzt, funktioniert die Auswertung der Daten auch ähnlich. Der analoge Ausgang liefert hier jedoch bis zu 10V , das heißt, dass die Spannung am Ausgang durch einen Spannungsteiler reduziert werden muss. Mit den Widerständen $R_1 = 68\text{k}\Omega$ und $R_2 = 33\text{k}\Omega$ ergibt sich hierbei durch Gleichung 2.9 folgende Maximalspannung, die am Widerstand R_2 abgegriffen werden kann:

$$U_2 = \frac{R_2}{R_1 + R_2} \cdot U_{max} = \frac{33\text{k}\Omega}{33\text{k}\Omega + 68\text{k}\Omega} \cdot 10\text{V} = 3.27\text{V} \quad (2.20)$$

Da maximal 3.3V am Analog-Digital-Wandler anliegen dürfen, eignet sich dieser Spannungsteiler somit sehr gut, um über R_2 die Ausgangsspannung abzulesen. Die restliche Schaltung ist in folgender Abbildung zu sehen:

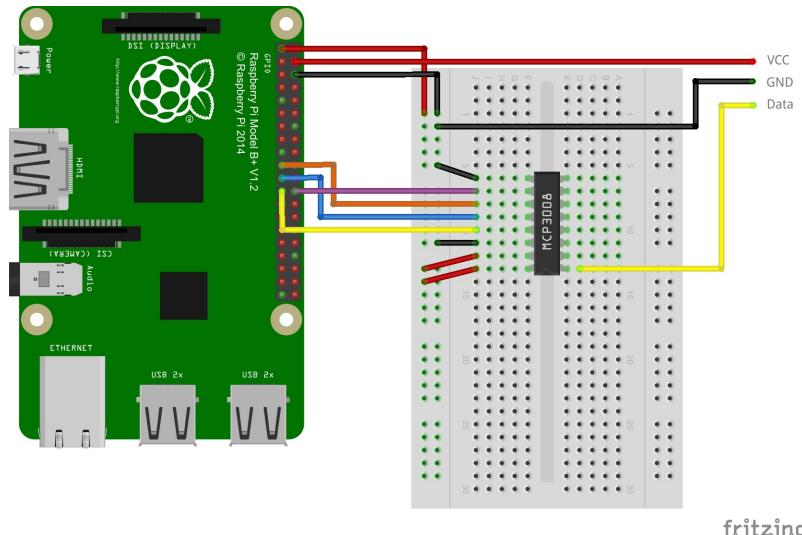


Abbildung 2.21: Schaltplan der Ausgangsspannung am Raspberry Pi[11]

2.2.3.3 Messwiederholungsrate

Laut Herstellerangabe besitzt der Sensor eine Messfrequenz von circa 100Hz . Nachdem der Schall jedoch etwa 3ms benötigt, um eine Distanz von einem Meter zu überbrücken, überlappen sich die einzelnen Messungen. Das heißt, es wird bereits eine weitere Messung gestartet, bevor die vorherige Messung beendet ist. Deswegen wird dann zuerst auf eine einzelne Messung zurückgerechnet und im Anschluss ein Mittelwert durch viele Messwerte berechnet. Letztendlich liefert der Analogausgang aktualisierte Ergebnisse mit einer Frequenz von 30Hz . Dies entspricht in etwa der bisherigen Messfrequenz und ist somit für diesen Anforderungsbereich ausreichend [12]

2.2.3.4 Messgenauigkeit/Reproduzierbarkeit

Um die Reproduzierbarkeit der Messwerte zu testen, wurde der Sensor auf ein Intervall von $30 - 60\text{cm}$ kalibriert. Dies entspricht auch dem Bereich, der nachher im Versuchsaufbau benötigt wird. Anschließend wurde im gesamten Messbereich in 5mm -Schritten eine Messung durchgeführt. Dies wurde insgesamt drei mal wiederholt, um die Statistik zu verbessern. Die Messwerte sind in folgendem Plot zu sehen:

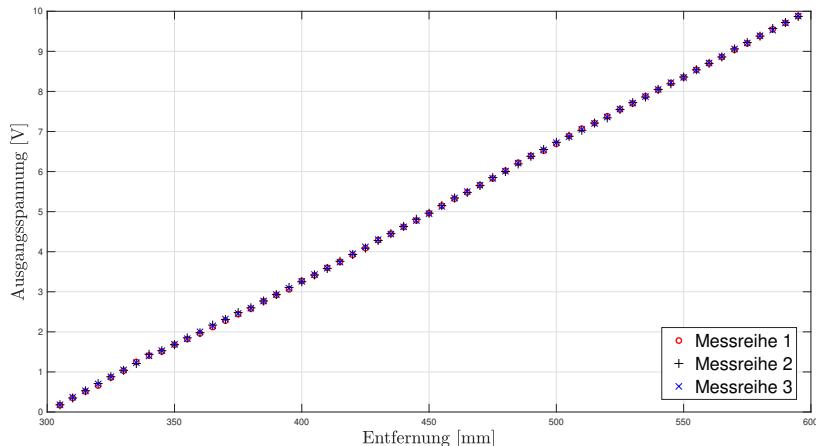


Abbildung 2.22: 3 Messreihen zur Ausgangsspannung des zweiten Ultraschallsensors in Abhängigkeit der Entfernung

Wie man sehr gut erkennen kann, liegen die drei Messpunkte pro Entfernung fast exakt aufeinander, trotz der Ungenauigkeiten durch das Ablesen der Entfernung auf der optischen Bank und der Ungenauigkeiten durch das Spannungsmessgerät. Dies

zeugt von einer sehr guten Reproduzierbarkeit der Messwerte. Da ein linearer Zusammenhang zwischen der Spannung und der Entfernung besteht, wurde anschließend eine Ausgleichsgerade durch die Messpunkte gelegt:

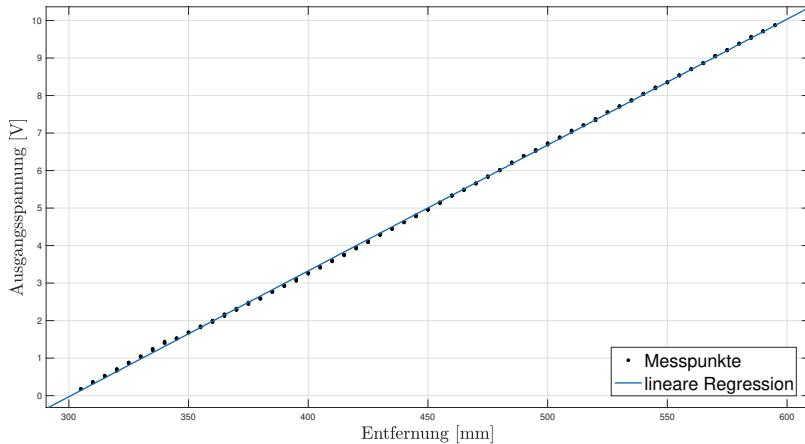


Abbildung 2.23: Linearer Fit durch die 3 Messreihen des Ultraschallsensors

Dabei lautet die Gleichung der Ausgleichsgerade in Abhängigkeit der Distanz D :

$$U(V) = D(\text{mm}) \cdot (0.03355 \pm 0.00008) \frac{V}{\text{mm}} - (10.09 \pm 0.04)V \quad (2.21)$$

Durch die geringen Unsicherheiten in der Gleichung lässt sich wiederum auf eine sehr hohe Reproduzierbarkeit und auch genaue Messung schließen. Das Bestimmtheitsmaß dieses linearen Fits liegt außerdem bei $R^2 = 99.98\%$, was diese Annahme noch zusätzlich bestätigt. Diese Gleichung muss aber natürlich nach jeder Kalibrierung neu erstellt werden, um die Spannung in eine Entfernung umzurechnen.

2.2.3.5 Fazit

Abschließend lässt sich also zu diesem Sensor feststellen, dass er sehr gut für die Anwendung im Praktikumsbetrieb geeignet ist. Einerseits liefert er eine relativ hohe Messrate und andererseits ist seine Reproduzierbarkeit und Genauigkeit sehr gut. Die Umrechnung von Spannung in die eigentlich gesuchte Entfernung muss dennoch im Nachhinein am Raspberry Pi erfolgen, was jedoch kein Problem darstellt. Aus diesen Gründen wurde beschlossen, diesen Sensor für den Versuch zu benutzen und einen Aufbau damit zu erstellen.

Kapitel 3

Bau des neuen Versuches

3.1 Hardware der Datenerfassung

Zur Aufnahme der Schwingung am Aufbau, wurde für den Sensor sowohl eine Halterung als auch ein Detektorschirm entworfen und aus Aluminium gefertigt. Da sich an dem Sensor selbst ein Gewinde befindet, kann dieser über zwei Muttern mit einer Gummidämpfung befestigt werden:



Abbildung 3.1: Waycon Ultraschallsensor [12]

Dieser wurde daraufhin mit folgender Halterung am Versuchsaufbau montiert (linke Abbildung). Zur Reflektion der Ultraschallwellen wurde außerdem auf dem Gleiter ein Detektorschirm befestigt (rechte Abbildung):

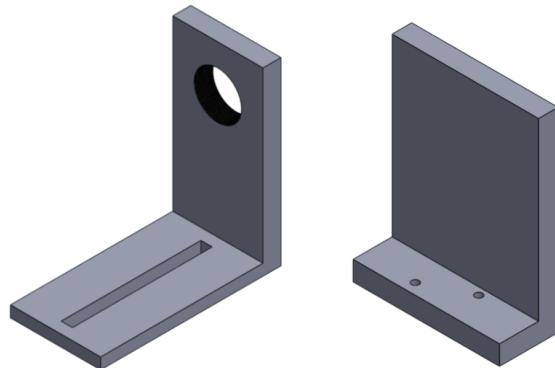


Abbildung 3.2: Sensorhalterung und Detektorschirm

Die genauen Maße sind im Anhang unter Kapitel D zu finden. Da bei der Überprüfung des Ultraschallsensors HC-SR04 im Kapitel 2.2.1.4 bereits festgestellt wurde, dass eine Fokussierung notwendig ist, wurde auch für diesen Sensor eine Fokussierung entworfen. Es wurde zuerst versucht, eine Messung ohne eine zusätzliche Fokussierung durchzuführen. Hierbei traten jedoch durch die Reflexion der Schallwellen an der Feder des Gleiters sehr viele Messfehler auf. Daraufhin wurde die folgende anschraubbare Fokussierung aus Aluminium gefertigt:



Abbildung 3.3: Fokussierung für den Ultraschallsensor

Um Reflexionen innerhalb des Rohres zu vermeiden, wurde dieses noch mit einer circa 2.5mm dicken Schaumstoffmatte auf der Innenseite verkleidet. Dies erhöhte die Fokussierung der Ultraschallwellen erheblich und es wurden keine Fehlmessungen mehr durch die Feder detektiert. Die technische Zeichnung dieses Bauteils ist ebenfalls im Anhang unter Kapitel D zu finden.

3.2 Hardware der Datenverarbeitung

Zur Auswertung der Daten wurde der Einplatinencomputer Raspberry Pi zusammen mit einem 2.8 Zoll Touchscreen verwendet. Da wie bereits erwähnt auch Wert auf die Didaktik gelegt werden soll, wurde vereinbart, dass die Kalibrierung des Sensors durch die Studenten selbst erfolgen sollen. Erfolgt die Erwähnung des Kalibriervorganges nur in der Versuchsanleitung, werden sich die Studenten kaum Gedanken über die Notwendigkeit und auch die Funktionsweise machen. Werden Sie jedoch selbst mit dieser Aufgabe gefordert, fördert dies das Verständnis und den Lerneffekt. Zur Kalibrierung muss der Kalibrier-Eingang des Sensors entweder mit der Betriebsspannung $+U_B$ oder der Masse GND verbunden werden. Durch das Verbinden mit der Betriebsspannung wird die maximale Messdistanz festgelegt, durch das Verbinden mit der Masse die minimale Messdistanz. Hierfür wird somit eine Relais-Schaltung benötigt. Nachdem die meisten Relais erst ab einer Nennspannung von 5V durchschalten, die GPIOs des Raspberry jedoch nur 3.3V liefern, muss zwischen diesen auch noch ein Transistor installiert werden. Um das Mikrocontrollerboard letztendlich noch galvanisch von dem Arbeitsstromkreis zu trennen, wird zwischen dem Raspberry Pi und der Schaltung noch ein Optokoppler positioniert. Da es eine solche Schaltung bereits fertig im Handel gibt, wurde diese nicht selbst konstruiert. Diese Schaltung wird durch den Raspberry angesteuert und ermöglicht somit eine Kalibrierung des Sensors über den Raspberry.

Für die Stromversorgung des Sensors wurde das Labornetzgerät PPS-13610 von Voltcraft verwendet. Dieses liefert 0 – 18V und wird bereits in dem Wärmeleitversuch verwendet. Durch die Verwendung des identischen Netzgerätes in beiden Versuchen, vereinfachen sich die Aufbauten erheblich, da nicht für jeden Versuch ein spezifisches Netzgerät angeschafft werden muss. Außerdem lassen sich die Netzteile bei einem Defekt einfach untereinander austauschen.

Die Datenübertragung selbst erfolgt, wie bereits unter Kapitel 2.2.3.2 erläutert, mithilfe eines Analog-Digital-Wandlers. Der gesamte Schaltplan ist im Anhang unter Kapitel C.1 zu finden. Für eine kompakte, robuste und übersichtliche Gestaltung, wurde mithilfe eines 3D-Druckers ein Gehäuse für die gesamte Schaltung inklusive Raspberry Pi und Touchscreen entworfen:

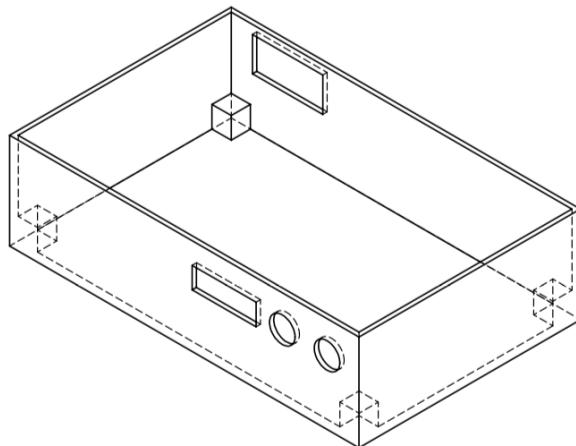


Abbildung 3.4: Box für die Schaltung des Sensors

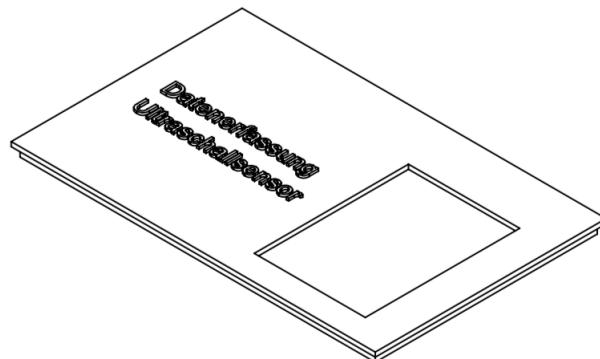


Abbildung 3.5: Deckel für die Schaltung des Sensors

Die runden Öffnungen in der Box dienen zum Anschließen der Bananenstecker, welche die Stromversorgung für den Sensor sicherstellen. Die beiden rechteckigen Öffnungen sind einerseits für den USB-Anschluss am Raspberry und andererseits für die Stromversorgung des Raspberries sowie die restlichen Kabel für den Sensor. Die gesamte Schaltung wurde auf einer Europlatine verlötet, auch der Raspberry Pi inklusive Touchscreen wurden ebenfalls auf dieser Platine festgeschraubt. Die Größe der Europlatine entspricht genau dem Innenmaß der Box. Durch die erhöhten

3.3 Software der Datenerfassung und Verarbeitung

Aufsätze in den Ecken liegt die Platine mit allen Anschlüssen jeweils passend an den entsprechenden Aussparungen. Die Öffnung im Deckel ist genau auf die Größe des Touchscreens abgestimmt. Somit ist die gesamte Schaltung durch die Box geschützt und nur der Display für die Studenten frei zugänglich. Der fertige Aufbau sieht dann folgendermaßen aus:

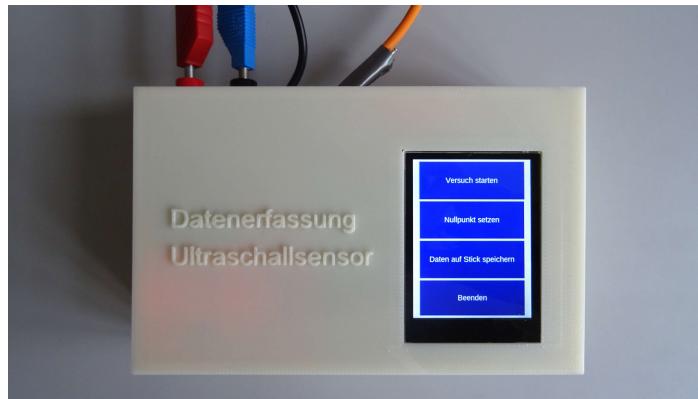


Abbildung 3.6: Fertiges Gehäuse für die Datenverarbeitung

3.3 Software der Datenerfassung und Verarbeitung

Um die Bedienung des Sensors mithilfe des Raspberrys möglichst einfach zu gestalten, wurde auf diesem mit Python eine grafische Oberfläche erstellt. Hierfür wurde PySimpleGUI verwendet, eine auf der Sprachanbindung Tkinter basierende Erweiterung.[13] Der Aufbau dieser Oberfläche ist in folgendem Diagramm dargestellt. Jeder Kasten ist hierbei ein eigenes Fenster. Die dick hinterlegte erste Zeile entspricht dem Namen des Fensters und die darauffolgenden Zeilen sind die auf dem Fenster vorhandenen Elemente:

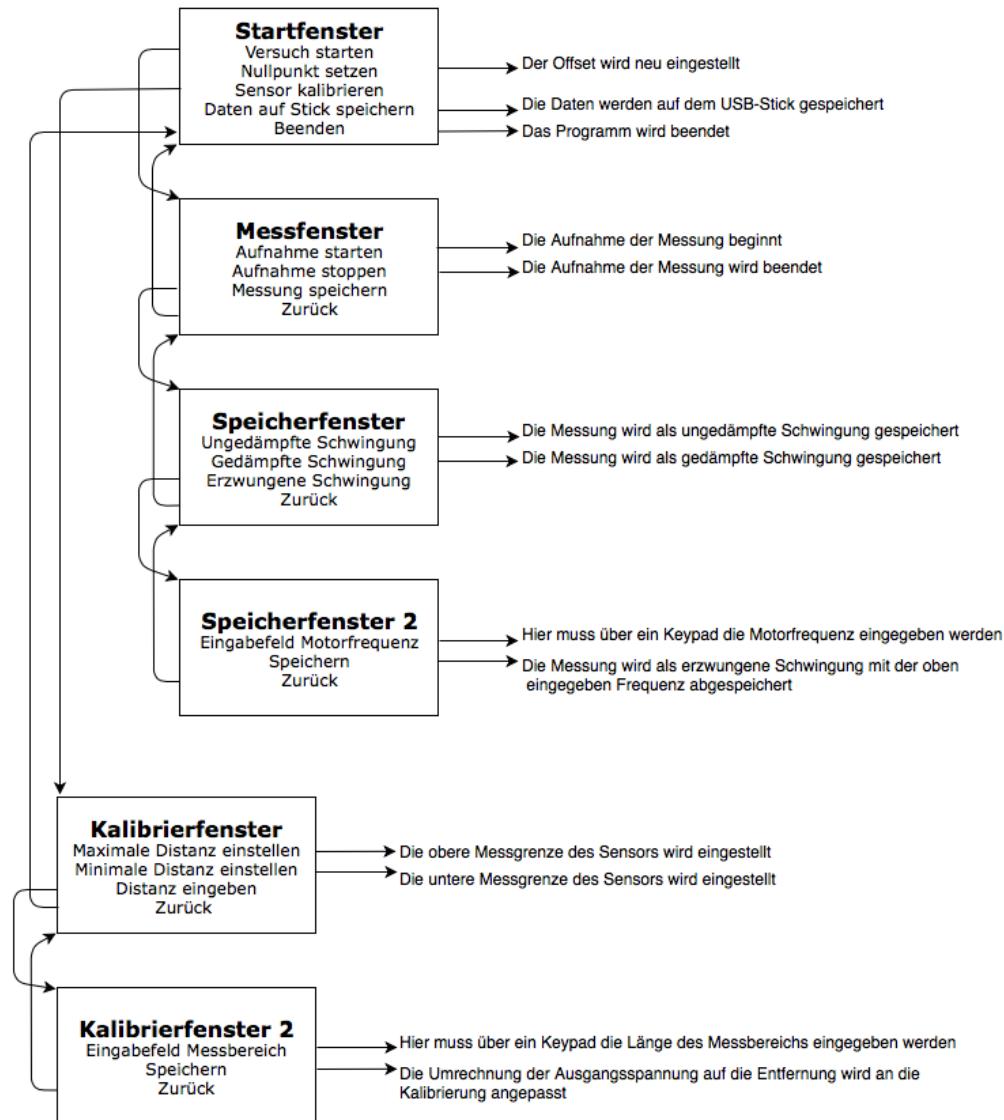


Abbildung 3.7: Flussdiagramm zum Aufbau der grafischen Oberfläche

Das reale Aussehen der Fenster auf dem Touchscreen ist in folgender Abbildung am Beispiel des Messfensters zu sehen:



Abbildung 3.8: Messfenster der grafischen Oberfläche

Hierbei stellt jedes Fenster eine Funktion in einer eigenen Datei dar, welche über einen Button aufgerufen werden kann. Mit jeder Speicherung einer Messung wird im Hintergrund mit der Programmbibliothek matplotlib eine Grafik erzeugt. Diese wird im Anschluss auf dem Raspberry selbst gesichert. Mit dem Button "Daten auf USB-Stick speichern", sucht der Raspberry nach einem angeschlossene USB-Stick und kopiert alle Grafiken auf diesen. Dadurch können die Abbildungen im Anschluss ausgedruckt und ausgewertet werden. Der Code für das gesamte Programm ist im Anhang unter B.4 zu finden.

3.4 Umbau der Bedienung des Motors

3.4.1 Umbau der Motorsteuerung

Die zweite Ummaßnahme nach der Datenerfassung ist die Steuerung und Anzeige der Motorfrequenz. Der bereits vorhandene Motor ist ein Gleichstrommotor der Firma Faulhaber. Bisher wurde dieser durch eine mit einem Gleichrichter umgewandelte Wechselspannung betrieben. An der Antriebsachse ist außerdem ein Generator befestigt. Durch diesen wurde die Regelung des Motors gesteuert und die tatsächliche Motorfrequenz an einem Display ausgegeben. Die Frequenz des Motors konnte hierbei in einem Intervall von 0.2Hz bis 2Hz durch eine Spannung von bis zu 6V eingestellt werden. Im Rahmen des Umbaus sollten sowohl die Spannungsquelle als auch die Anzeige der Frequenz erneuert werden.

Für eine Konsistenz mit den anderen Aufbauten wurde als neue Spannungsquelle ebenfalls das Netzgerät PPS-13610 von Voltcraft verwendet. Dieses besitzt eine maximale Ausgangsspannung von 18V und eine Remote Control Steuerung. Diese funktioniert mit 0 – 5V. 0V am Remote Control Eingang entsprechen somit 0V am Ausgang des Netzgeräts und 5V entsprechen der Maximalspannung von 18V. Da für unseren Zweck eine Steuerung der Ausgangsspannung von 0 – 6V ausreichend ist, wurde mithilfe eines Spannungsteilers die maximal am Remote Control anliegende Spannung auf 1.67V reduziert. Dies ermöglicht eine Steuerung der Ausgangsspannung von 0 – 6V. Zur Steuerung selbst wurde ein 10-Gang-Potentiometer verwendet, um eine möglichst genaue Justierung der Frequenz zu ermöglichen. Der Schaltplan hierzu ist im Anhang unter C.3 beigefügt.

3.4.2 Umbau der Frequenzbestimmung

Um die aktuelle Drehfrequenz des Motors mithilfe des Generators auslesen zu können, wurde wiederum ein Raspberry Pi verwendet. Das Prinzip ist hierbei ähnlich dem Auslesen des Ultraschallsensors. Die analoge Ausgangsspannung des Generators wird mithilfe eines Spannungsteilers reduziert, dann über einen Analog-Digital-Wandler in ein digitales Signal umgewandelt und letztendlich am Raspberry ausgelesen. Der Schaltplan hierzu ist im Anhang unter C.2 beigefügt.

Damit dieser Aufbau ebenfalls kompakt und übersichtlich ist, wurde auch hier alles auf einer Europlatine befestigt und diese in einer 3D-gedruckten Box verbaut. Der Unterschied zum vorherigen Gehäuse ist diesmal, dass neben dem Display auch noch ein Potentiometer zur Motorsteuerung in dem Deckel zu finden ist:

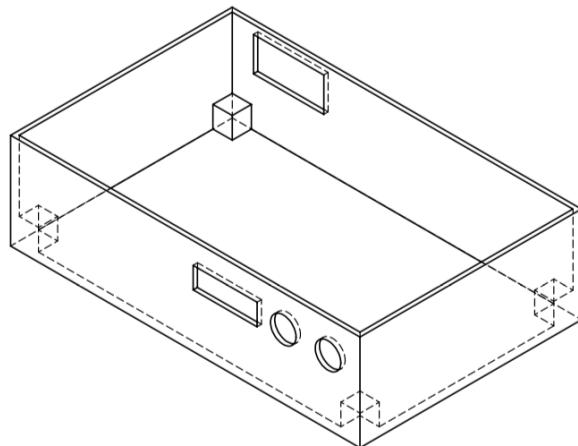


Abbildung 3.9: Box für die Schaltung der Motorsteuerung

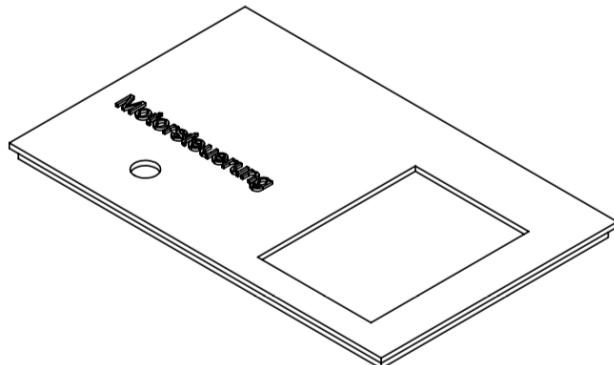


Abbildung 3.10: Deckel für die Schaltung der Motorsteuerung

Um eine möglichst genaue Umrechnung des digitalen Signals in die Frequenz zu ermöglichen, wurde das digitale Signal in Abhängigkeit der Frequenz bestimmt und anschließend eine Ausgleichsgerade durch diese Messwerte gelegt. Die periodische, äußere Kraft auf den Gleiter wird durch einen Kurbelantrieb vom Motor erzeugt. Durch diesen Kurbelantrieb ist die Belastung des Motors jedoch nicht konstant. Die Frequenz der Schwingung unterlag deshalb gewissen Schwankungen. Um das Ablesen des digitalen Signals B , also die 0 bis 1023 Einheiten, zu vereinfachen, wurde

daher immer auf die nächste Zehnerpotenz gerundet. Dieser Wert wurde dann am Raspberry Pi ausgegeben. Die Bestimmung der Frequenz des Motors wurde mit Hilfe eines Laser-Frequenzmessers durchgeführt. Dieser hat eine Ungenauigkeit von $\pm 0.05\%$ der Frequenz, jedoch immer mindestens $1U/min$. Die Messergebnisse inklusive einer linearen Regression sind in folgender Abbildung dargestellt. Nachdem die Unsicherheiten der Messwerte so klein sind und in der grafischen Darstellung nicht wirklich erkennbar sind, wurden sie aufgrund der Übersichtlichkeit in dem Plot nicht dargestellt.

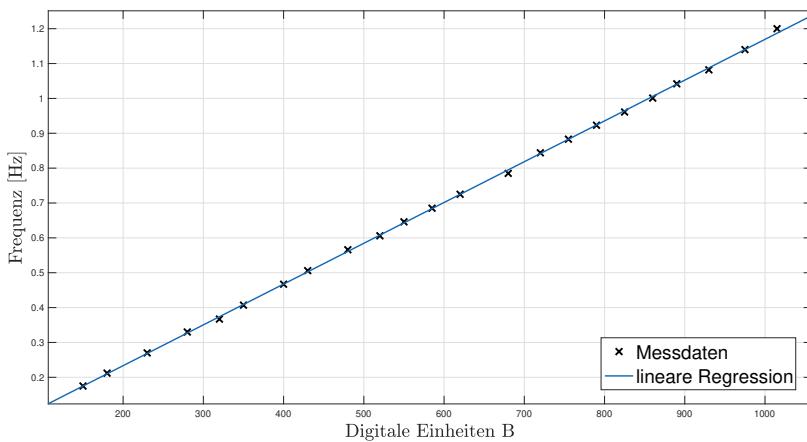


Abbildung 3.11: Motorfrequenz in Abhängigkeit der digitalen Einheiten inklusive einer linearen Regression

Die Ausgleichsgerade hat hierbei eine Form von:

$$\text{Frequenz}(Hz) = (0.00117 \pm 0.00001)Hz \cdot B - (0.00061 \pm 0.00502)Hz \quad (3.1)$$

Das Bestimmtheitsmaß dieser linearen Regression liegt bei $R^2 = 99.98\%$. Die Ergebnisse bestätigen, dass die Messung der Frequenz mit dem Laserfrequenzmesser und dem Raspberry Pi somit sehr gut funktionieren. Diese Funktion wurde anschließend mit einem Faktor von 1000 in die grafische Oberfläche auf dem Raspberry Pi eingepflegt. Somit zeigt dieser nun die Frequenz in [mHz] auf dem Display an. Die fertige Box zur Steuerung und Auslese des Motors ist in folgender Grafik zu sehen. Mithilfe des Potentiometers kann die Frequenz des Motors verändert und dabei direkt am Display daneben ausgelesen werden:

3.4 Umbau der Bedienung des Motors



Abbildung 3.12: Fertiges Gehäuse für die Motorsteuerung

Kapitel 4

Überlegungen zur Didaktik

Mit dem Versuch 'Schwingung und Resonanz' soll den Studenten ein anschauliches und praktisches Beispiel für harmonische, gedämpfte und erzwungene Schwingungen sowie für den Begriff 'Resonanz' gegeben werden. Hierbei sind natürlich jede Menge verschiedener Versuche möglich und auch diverse Variationen dieser. Damit jedoch auch einer großen Anzahl von Studenten diese Möglichkeit gegeben werden kann, muss der zeitliche Rahmen begrenzt werden. Man entschied sich somit auf eine Dauer von drei bis vier Stunden pro Versuch. Um den Studierenden in diesem Zeitfenster einen Gesamtüberblick zur genannten Thematik zu vermitteln, wurde anschließend ein Versuch mit drei Variationen konstruiert. Hierbei werden Messdaten mithilfe eines Gleiters auf einer Luftkissenbahn erfasst. Dieser Aufbau eignet sich sehr gut, da die Bewegung des Gleiters auch mit dem Auge zu sehen ist. Dadurch bekommen die Studierenden bereits zu Beginn ein Gefühl für die Größenordnungen und dem Grundprinzip einer Schwingung. Mit diesem Apparat sollen die Studenten anhand von einigen Messdurchgängen spezifische Werte, wie z.B. die Eigenfrequenz oder die Dämpfung, für ein solches System bestimmen. Bei der ersten Variante wird eine ungedämpfte Schwingung analysiert, um zuerst eine grobe Einschätzung zu bekommen. Die darauffolgenden Varianten befassen sich dann mit einer Dämpfung und externen Kraft, sowie mit der Resonanz. Dadurch werden alle großen Konzepte im Bereich Schwingung abgedeckt.

Danksagung

Ich möchte mich bei meinem Betreuer Rainer Stoepler für die gute Unterstützung während der Arbeit bedanken, sowie bei Thomas Deuschele für den Zusammenbau und die Fertigung der Komponenten. Außerdem möchte ich mich bei Dr. Christina Scharnagel und bei Thomas Zeitzler für die Unterstützung in Weihenstephan bedanken.

Anhang A

Abkürzungsverzeichnis

IR-Sensor Infrarotsensor

GPIO Allgemeiner Ausdruck der General Purpose Input/Output Anschlüsse des Raspberrys

US-HC-SR04 Ultraschallsensor vom Typ HC-SR04 der Firma OSEPP Electronics

IR-GP2Y0A21YK0F Infrarotsensor GP2Y0A21YK0F der Firma Sharp

US-UFA-1500 Eine Serie der Ultraschallsensoren der Firma WayCon

Anhang B

Quellcode

B.1 Klasse für den Analog Digital Wandler MCP3008

```
from spidev import SpiDev

#Klasse erstellen fuer den ADC
class MCP3008:
    def __init__(self, bus=0, device=1):
        self.bus, self.device = bus, device
        self.spi = SpiDev()
        self.open()

    def open(self):
        self.spi.open(self.bus, self.device)

    def read(self, channel=0):
        adc = self.spi.xfer2([1, (8 + channel) << 4, 0])
        data = ((adc[1] & 3) << 8) + adc[2]
        return data

    def close(self):
        self.spi.close()
```

[14]

B.2 Ansteuerung des Ultraschallsensors HC-SR04

```
# Bibliotheken einbinden
import RPi.GPIO as GPIO
import time

# GPIO Modus (BOARD / BCM)
GPIO.setmode(GPIO.BCM)

# GPIO Pins zuweisen
GPIO_TRIGGER = 21
GPIO_ECHO = 26

# Richtung der GPIO-Pins festlegen (IN / OUT)
GPIO.setup(GPIO_TRIGGER, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)

def distanz():
    # setze Trigger auf HIGH
    GPIO.output(GPIO_TRIGGER, True)

    # setze Trigger nach 0.01ms auf LOW
    time.sleep(0.00001)
    GPIO.output(GPIO_TRIGGER, False)

    StartZeit = time.time()
    StopZeit = time.time()

    # speichere Startzeit
    while GPIO.input(GPIO_ECHO) == 0:
        StartZeit = time.time()

    # speichere Ankunftszeit
    while GPIO.input(GPIO_ECHO) == 1:
        StopZeit = time.time()

    # Zeit Differenz zwischen Start und Ankunft
    TimeElapsed = StopZeit - StartZeit
```

```
# mit der Schallgeschwindigkeit (34300 cm/s)
# → multiplizieren
# und durch 2 teilen, da hin und zurueck
distanz = (TimeElapsed * 34300) / 2

return distanz

if __name__ == '__main__':
    try:
        while True:
            abstand = distanz()
            print ("Gemessene Entfernung = %.1f cm" %
                  abstand)
            time.sleep(1)

        # Beim Abbruch durch STRG+C resetten
    except KeyboardInterrupt:
        print("Messung vom User gestoppt")
        GPIO.cleanup()
```

[15]

B.3 Auslesen des Infrarotsensors GP2Y0A21YK0F

```
from MCP3008 import MCP3008

adc = MCP3008()

try:
    while True:
        value = adc.read(channel=0)
        print("Anliegende_Spannung: %.2f" % (value / 1023.0
                                                * 3.3))
        time.sleep(0.1)

    # Beim Abbruch durch STRG+C resetten
except KeyboardInterrupt:
    print("Messung_vom_User_gestoppt")
    GPIO.cleanup()
```

[14]

B.4 Grafische Oberfläche für den Sensor

B.4.1 Startfenster

```
import PySimpleGUI as sg #importiert PySimpleGUI fuer  
    → grafische Oberflaeche  
import time #wird fuer die Zeitmessung benoetigt  
import multiprocessing #wird fuer die Multiprozesse  
    → benoetigt  
import numpy as np #wird zur Verarbeitung der Daten  
    → benoetigt  
import matplotlib.pyplot as plt #wird fuer die Erstellung  
    → der Graphen benoetigt  
import shutil #ermoeglicht Daten-Uebertragung auf Stick  
import os #ermoeglicht Operationen auf Systemebene  
from MCP3008 import MCP3008 #importiert Klasse fuer den ADC  
from swi4messen import windowmessen #importiert Messfenster  
from swi4kalibrieren import windowkalibrieren #importiert  
    → Kalibrierfenster  
import RPi.GPIO as GPIO #wird zur Ansteuerung der GPIOs  
    → benoetigt  
GPIO.setwarnings(False)  
  
#GPIOs fuer Kalibrieren des Sensors festlegen:  
GPIO.setmode(GPIO.BCM)  
GPIO.setup(26, GPIO.OUT)  
GPIO.setup(21, GPIO.OUT)  
GPIO.output(26, GPIO.LOW)  
GPIO.output(21, GPIO.LOW)  
  
adc = MCP3008()  
  
sumoffset = 0  
  
  
  
#Fensterlayout wird erstellt:  
layout = [[ sg.Button("Versuch_starten", font=("Helvetica"  
    → ,23), key="windmessen", size=(100, 3))],
```

```
[ sg.Button("Nullpunkt_setzen", font=("Helvetica",
    ↪ 23), key="setoffset", size=(100, 3))],
[ sg.Button("Daten_auf_Stick_speichern", font=(
    ↪ Helvetica", 23), key = "savedata", size=(100,
    ↪ 3))],
[ sg.Button("Sensor_kalibrieren", key="kali", font
    ↪ =("Helvetica", 23), size=(100, 3))],
    [sg.Button("Beenden", key="exit", font=(
        ↪ Helvetica", 23), size=(100, 3))]]
```

#Fenster wird erstellt:

```
window = sg.Window("home", grab_anywhere=False, location=(1,
    ↪ 1), size=(480, 640)).Layout(layout).Finalize()
```

#Fenster wird maximiert

```
window.Maximize()
```

#Schleife um das Fenster zu lesen:

```
while True:
    button, values = window.Read()
    if button == "windmessen": #oeffnet das Messfenster
        windowmessen()
    if button == "setoffset": #Der Offset wird
        ↪ festgelegt
        for i in range(5): #Mittelwert ueber 5
            ↪ Messungen
            m = adc.read(channel = 0) #Sensor
            ↪ wird ausgelesen
            sumoffset = sumoffset + m
            time.sleep(0.1)
        offset = sumoffset / 5
        sg.Popup("Der_Offset_wurde_neu_eingestellt",
            ↪ no_titlebar=True, font=("Helvetica",
            ↪ 21), background_color=("blue"))
        file_out = open("Offset.txt", "w")
        file_out.write(str(offset)) #Offset wird in
            ↪ Datei abgespeichert
        file_out.close()
    if button == "savedata": #Daten werden auf USB Stick
        ↪ uebertragen
```

```

ordner=[]
for r, d, f in os.walk("/media/pi/"):
    ↪ Verzeichnis des Sticks wird gesucht
    ordner.append(d[0])
    break
try: #Test, ob auf Stick noch alte Messdaten
    ↪ vorhanden, falls ja, dann loeschen
    shutil.rmtree("/media/pi/" + ordner
    ↪ [0] + "/messdaten/")
except OSError:
    pass
shutil.copytree("/home/pi/messergebnisse/",
    ↪ "/media/pi/" + ordner[0] + "/messdaten/") #
    ↪ kopiert neue Messdaten auf Stick
shutil.rmtree("/home/pi/messergebnisse/") #
    ↪ Loescht Messdaten der Studenten auf
    ↪ Raspberry
os.makedirs("/home/pi/messergebnisse") #
    ↪ Legt neues Verzeichnis fuer naechste
    ↪ Studenten auf Raspberry an
sg.Popup("Die Daten wurden uebertragen",
    ↪ no_titlebar=True, font=("Helvetica",
    ↪ 21), background_color=("blue"))
if button == "kali": #oeffnet das Kalibrierfenster
    windowkalibrieren()
if button is None or button == "exit": #Beendet das
    ↪ Programm
    break
window.Close()

```

B.4.2 Fenster zum Aufnehmen der Messung

```
import PySimpleGUI as sg
import time
import multiprocessing
import numpy as np
import matplotlib.pyplot as plt
from MCP3008 import MCP3008
from swi4speichern import windowsave

def messen(recv_end): #Funktion zum Erfassen der Daten
    adc = MCP3008()
    offset = np.loadtxt("Offset.txt") #liest Offset aus
    ↪ Datei
    offset = float(offset)
    digitpermm = np.loadtxt("Kalibrierung.txt") #liest
    ↪ Umrechnungsfaktor fuer Entfernung aus Datei
    digitpermm = float(digitpermm)
    startzeit = time.time() #setzt Startzeit fest
    file_out = open("Datentabelle.txt", "w")
    while True:
        if recv_end.poll(): #wird zum Beenden der
            ↪ Messung benoetigt
            a = recv_end.recv() #pipe wird
            ↪ wieder geleert
            file_out.close()
            break
        else:
            zeitsum = 0
            distanzsum = 0
            for i in range(10): #Schleife fuer
                ↪ Mittelwert ueber Messwerte
                valuen = adc.read(channel =
                    ↪ 0) #liest den ADC aus
                endzeit = time.time()
                zeit = endzeit - startzeit #
                ↪ Berechnet Zeitdifferenz
                valuecorrected = valuen -
```

```

    ↵ offset #Korrigert
    ↵ Messwert mit Offset
distanz = valuecorrected /
    ↵ digitperm #Umrechnun
    ↵ in [mm]
distanzsum = distanzsum +
    ↵ distanz
zeitsum = zeitsum + zeit
time.sleep(0.003)
finaldistanz = distanzsum / 10
finalzeit = zeitsum / 10
file_out.write("%2f,%8.4f\n" %
    ↵ finaldistanz, finalzeit)) #
    ↵ Schreibt Messwerte in Datei

def windowmessen():
    recv_end, send_end = multiprocessing.Pipe()
    layout = [[sg.Button("Aufnahme_starten", font=(
        ↵ "Helvetica", 23), key="start", size=(100, 4))
    ↵ ],
        [sg.Button("Aufnahme_stoppen",
            ↵ font=("Helvetica", 23), key=(
                ↵ "stop"), size=(100, 4)),
        [sg.Button("Messung_speichern",
            ↵ font=("Helvetica", 23), key=(
                ↵ "save"), size=(100, 4)),
        [sg.Button("Zurueck", font=(
            ↵ "Helvetica", 23), key="wind0"
            ↵ ), size=(100, 4)]]]
    window = sg.Window("measure_window", grab_anywhere=
        ↵ False, location = (1, 1), size = (480, 640)).
        ↵ Layout(layout).Finalize()
    window.Maximize()
    while True:
        button, values = window.Read()
        if button == "start": #startet die Messung
            ↵ mit einem Multiprozess

```

```
p1=multiprocessing.Process(target =
    ↪ messen , args=(recv_end ,) )
p1.start()
sg.Popup("Messung_wurde_gestartet",
    ↪ no_titlebar=True , font=(
    ↪ Helvetica" , 21) ,
    ↪ background_color=("blue"))
if button == "stop": #Beendet die Messung
    ↪ durch das pipe
        send_end.send("0")
        time.sleep(0.1)
        sg.Popup("Messung_wurde_gestoppt",
            ↪ nun_bitte_Speichern!" ,
            ↪ no_titlebar=True , line_width
            ↪ =(40) , background_color = ("blue
            ↪ ") , font=("Helvetica" , 18))
if button == "save": #Oeffnet das
    ↪ Speicherfenster
        windowsave()
if button == "wind0": #Schliesst das Fenster
    break
window.Close()
```

B.4.3 Fenster zum Speichern der freien Schwingungen

```

import PySimpleGUI as sg
import time
import multiprocessing
import numpy as np
import matplotlib.pyplot as plt
from swi4speichern2 import windowsave2
import os

def save(): #Funktion zum Erzeugen/Speichern der Graphen
    ywerte = np.genfromtxt("Datentabelle.txt", usecols
        ↪ =(0)) #importiert Messwerte
    xzeit = np.genfromtxt("Datentabelle.txt", usecols
        ↪ =(1)) #importiert die Zeitwerte
    plt.figure(figsize=(20.48, 7.68)) #setzt Groesse der
        ↪ Grafik
    plt.ylabel("Distanz_in_[mm]") #Achsenbeschriftung
    plt.xlabel("Zeit_in_[s]")
    plt.grid(True) #Fuegt ein Gitter ein
    plt.plot(xzeit, ywerte, "b.") #Erzeugt die Grafik

def windowsave():
    layout = [[sg.Text("Was war dies fuer", font=("
        ↪ Helvetica", 39))],
              [sg.Text("eine Messung?", font=(
        ↪ Helvetica", 39))],
              [sg.Button("Ungedaempfte",
        ↪ Schwingung", font=("Helvetica
        ↪ ", 23), key="save1", size
        ↪ =(100, 3))],
              [sg.Button("Gedaempfte_Schwingung"
        ↪ , font=("Helvetica", 23), key=
        ↪ ="save2", size=(100, 3))],
              [sg.Button("Erzwungene_Schwingung"
        ↪ , font=("Helvetica", 23), key=
        ↪ "save3", size=(100, 3))],
              [sg.Button("Zurueck", font="

```

```
    ↪ Helvetica" , 23) , key="windhome
    ↪ " , size=(100, 2))]]
window = sg.Window("save_window" , grab_anywhere=
    ↪ False , location=(1, 1) , size=(480, 640)).Layout
    ↪ (layout).Finalize()
window.Maximize()
while True:
    button , values = window.Read()
    if button == "save1": #Speichert Schwingung
        ↪ als ungedaempfte Schwingung
        save()
        plt.title("Ungedaempfte_Schwingung")
        ↪ #setzt den Titel der Grafik
        plt.savefig("/home/pi/messergebnisse
        ↪ /aufgabe1.png") #Speichert die
        ↪ Grafik
        sg.Popup("Diese_Messung_wurde_
        ↪ gespeichert" , no_titlebar=True ,
        ↪ font=("Helvetica" , 21) ,
        ↪ background_color=("blue"))
    if button == "save2": #Speichert Schwingung
        ↪ als gedaempfte Schwingung
        save()
        plt.title("Gedaempfte_Schwingung") #
        ↪ setzt den Titel der Grafik
        plt.savefig("/home/pi/messergebnisse
        ↪ /aufgabe2.png") #Speichert die
        ↪ Grafik
        sg.Popup("Diese_Messung_wurde_
        ↪ gespeichert" , no_titlebar=True ,
        ↪ font=("Helvetica" , 21) ,
        ↪ background_color=("blue"))
    if button == "save3": #oeffnet zweites
        ↪ Speicherfenster
        windowsave2()
    if button == "windhome": #Schliesst Fenster
        os.remove("Datentabelle.txt") #
        ↪ Loescht Daten fuer naechste
        ↪ Aufnahme
    break
window.Close()
```

B.4.4 Fenster zum Speichern der erzwungenen Schwingungen

```

import PySimpleGUI as sg
import time
import multiprocessing
import numpy as np
import matplotlib.pyplot as plt

def save(): #Funktion zum Erzeugen/Speichern der Graphen
    ywerte = np.genfromtxt("Datentabelle.txt", usecols
                           ↪ =(0)) #importiert Messwerte
    xzeit = np.genfromtxt("Datentabelle.txt", usecols
                           ↪ =(1)) #importiert die Zeitwerte
    plt.figure(figsize=(20.48, 7.68)) #setzt Groesse der
                           ↪ Grafik
    plt.ylabel("Distanz in [mm] ") #Achsenbeschriftung
    plt.xlabel("Zeit in [s]")
    plt.grid(True) #Fuegt ein Gitter ein
    plt.plot(xzeit, ywerte, "b.") #Erzeugt die Grafik

def windowsave2():
    layout = [[sg.Text("Motorfrequenz in [mHz] eingeben",
                      ↪ font=("Helvetica", 22))],
              [sg.Input(size=(100, 26), font=("Helvetica",
                                             ↪ 22), do_not_clear=True, justification=""
                                             ↪ center", key="input")],
              [sg.Button("1", font=("Helvetica", 31)), sg.
               ↪ Button("2", font=("Helvetica", 31)), sg.
               ↪ Button("3", font=("Helvetica", 31))],
              [sg.Button("4", font=("Helvetica", 31)), sg.
               ↪ Button("5", font=("Helvetica", 31)), sg.
               ↪ Button("6", font=("Helvetica", 31))],
              [sg.Button("7", font=("Helvetica", 31)), sg.
               ↪ Button("8", font=("Helvetica", 31)), sg.
               ↪ Button("9", font=("Helvetica", 31))],
              [sg.Button("Save", key="save", font=(
                  ↪ Helvetica", 29)), sg.Button("0", font=""

```

```
    ↪ Helvetica", 31)), sg.Button("Clear", font
    ↪ =("Helvetica", 27))],
[sg.Button("Zurueck", font=("Helvetica", 23),
    ↪ key="windsave", size=(100, 2))]]
window = sg.Window("save_window2", grab_anywhere=False,
    ↪ location=(1, 1), size=(480, 640),
    ↪ default_button_element_size=(5, 2),
    ↪ auto_size_buttons=False).Layout(layout).Finalize()
window.Maximize()
keys_entered = ""
while True:
    button, values = window.Read()
    if button in "1234567890": #Fuegt Werte von Keypad
        ↪ in Liste dazu
        keys_entered = values["input"]
        keys_entered += button
    if button == "save": #Speichert Messung mit
        ↪ Motorfrequenz
        keys_entered = values["input"]
        save()
        title3 = "Erzwungene Schwingung mit einer "
        ↪ Frequenz von " + str(keys_entered) + " [mHz]"
        ↪ #Titel der Grafik
        name3 = "/home/pi/messergebnisse/aufgabe3mit" +
            ↪ str(keys_entered) + "mHz.png" #Dateiname der
            ↪ Grafik
        plt.title(title3) #setzt Titel
        plt.savefig(name3) #speichert Grafik unter
            ↪ gegebenen Dateinamen
        sg.Popup("Diese Messung wurde gespeichert",
            ↪ no_titlebar=True, font=("Helvetica", 21),
            ↪ background_color=("blue"))
    if button == "Clear": #Input wird geloescht
        keys_entered = ''
    if button == "windsave": #schliesst Fenster
        break
window.Element('input').Update(keys_entered) #
    ↪ aktualisiert das Fenster fortwaehrend
window.Close()
```

B.4.5 Fenster zum Kalibrieren des Sensors

```

import PySimpleGUI as sg
import time
import multiprocessing
import numpy as np
import RPi.GPIO as GPIO
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(26, GPIO.OUT)
GPIO.setup(21, GPIO.OUT)
from swi4kalibrieren2 import windowkalibrieren2

def windowkalibrieren():
    layout = [[sg.Text("Maximale_Messdistanz_festlegen :",
                      font=("Helvetica", 22)),
               sg.Button("Kalibrieren", font=("Helvetica", 25),
                         key="kalimax", size=(100, 3))],
              [sg.Text("Minimale_Messdistanz_festlegen :",
                      font=("Helvetica", 22)),
               sg.Button("Kalibrieren", font=("Helvetica", 25),
                         key="kalimin", size=(100, 3))],
              [sg.Button("Messbereich_angeben", font=("Helvetica", 25),
                         key="windkalibrieren", size=(100, 3))],
              [sg.Button("Zurueck", font=("Helvetica", 25),
                         key="windhome", size=(100, 3))]]
    window = sg.Window("kalibrieren_window", grab_anywhere=False,
                       location=(1, 1), size=(480, 640)).Layout(
        layout).Finalize()
    window.Maximize()
    while True:
        button, values = window.Read()
        if button == "kalimax": #Maximale Messdistanz wird
            kalibriert
            GPIO.output(26, GPIO.HIGH)
            time.sleep(1)
            GPIO.output(26, GPIO.LOW)
            sg.Popup("Die_maximale_Distanz_wurde_eingestellt")

```

```
    ↵ " , no_titlebar=True , font=("Helvetica" , 21)
    ↵ , background_color=("blue"))
if button == "kalimin": #Minimale Messdistanz wird
    ↵ kalibriert
    GPIO.output(21 , GPIO.HIGH)
    time.sleep(1)
    GPIO.output(21 , GPIO.LOW)
    sg.Popup("Die_minimale_Distanz_wurde_eingestellt"
    ↵ " , no_titlebar=True , font=("Helvetica" , 21)
    ↵ , background_color=("blue"))
if button == "windkalibrieren": #oeffnet zweites
    ↵ Kalibrierfenster
    windowkalibrieren2()
if button == "windhome": #Schliesst Fenster
    break
window.Close()
```

B.4.6 Fenster zur Eingabe des Messbereiches

```

import PySimpleGUI as sg
import time
import multiprocessing
import numpy as np
import matplotlib.pyplot as plt

def windowkalibrieren2():
    layout = [[sg.Text("Messbereich in [mm] eingeben", font
        ↪ =("Helvetica", 22))],
              [sg.Input(size=(100, 26), font=("Helvetica",
                ↪ 22), do_not_clear=True, justification="
                  ↪ center", key="input")],
              [sg.Button("1", font=("Helvetica", 31)), sg.
                ↪ Button("2", font=("Helvetica", 31)), sg.
                ↪ Button("3", font=("Helvetica", 31))],
              [sg.Button("4", font=("Helvetica", 31)), sg.
                ↪ Button("5", font=("Helvetica", 31)), sg.
                ↪ Button("6", font=("Helvetica", 31))],
              [sg.Button("7", font=("Helvetica", 31)), sg.
                ↪ Button("8", font=("Helvetica", 31)), sg.
                ↪ Button("9", font=("Helvetica", 31))],
              [sg.Button("Save", key="save", font=(
                ↪ Helvetica", 29)), sg.Button("0", font=(
                  ↪ Helvetica", 31)), sg.Button("Clear", font
                    ↪ =("Helvetica", 27))],
              [sg.Button("Zurueck", font=("Helvetica", 23),
                ↪ key="windkalibrieren", size=(100, 2))]]
    window = sg.Window("kalibrieren_window2", grab_anywhere=
        ↪ False, location=(1, 1), size=(480, 640),
        ↪ default_button_element_size=(5, 2),
        ↪ auto_size_buttons=False).Layout(layout).Finalize()
    window.Maximize()
    keys_entered = ""
    while True:
        button, values = window.Read()

```

```
if button in "1234567890": #Fuegt Werte von Keypad
    ↪ in Liste dazu
    keys_entered = values["input"]
    keys_entered += button
if button == "save": #Speichert den Messbereich
    keys_entered = values["input"]
    distanz = str(keys_entered)
    digitperm = 1023 / float(distanz) #
        ↪ Umrechnungsfaktor fuer Entfernung wird
        ↪ berechnet
    file_out = open("Kalibrierung.txt", "w")
    file_out.write(str(digitperm)) #
        ↪ Umrechnungsfaktor wird in Datei
        ↪ abgespeichert
    file_out.close()
    sg.Popup("Die Distanz wurde eingestellt",
        ↪ no_titlebar=True, font=("Helvetica", 21),
        ↪ background_color=("blue"))
if button == "Clear": #Loescht den Input
    keys_entered =
if button == "windkalibrieren": #Schliesst das
    ↪ Fenster
    break
window.Element('input').Update(keys_entered)
window.Close()
```

B.5 Grafische Oberfläche für die Motorsteuerung

```

import PySimpleGUI as sg      #importiert Modul fuer GUI
from MCP3008 import MCP3008 #importiert Klasse fuer den ADC
import time
import math

adc = MCP3008()

#Fensterlayout wird erstellt:
layout =[[sg.Text("Motorfrequenz", font=("Helvetica",35),
    ↪ justification = "center")],
    [sg.Text("in [mHz]", font=("Helvetica",35),
    ↪ justification = "center")],
    [sg.Text("", font=("Helvetica", 40), justification =
    ↪ "center", key="_output_", size=(100, 4))],
    [sg.Button("Beenden", font=("Helvetica",23), size
    ↪ =(100, 4))]]

#Fenster wird erstellt:
window = sg.Window("enginemwindow", grab_anywhere=False,
    ↪ location=(1, 1), size=(480, 640)).Layout(layout).
    ↪ Finalize()

#Fenster wird maximiert:
window.Maximize()

#Schleife um das Fenster zu lesen:
while True:
    button1, values1 = window.Read(0)
    valuerntsum150=0
    for i in range(150):  #Schleife fuer Mittelwert ueber
        ↪ Motorfrequenz
        valuernt = adc.read(channel=0)    #Sensor
        ↪ auslesen
        valuerntsum150 = valuerntsum150 + valuernt
        time.sleep(0.005)
    valuerntsum = valuerntsum150 /150
    frequency = valuerntsum * 0.00117 - 0.0006098 #
        ↪ Umrechnung digitaler Input in Frequenz

```

Anhang B Quellcode

```
frequencymhz = frequency * 1000 #Umrechnung von Hz
    ↪ in mHz
frequencymhz = int(frequencymhz)
frequencymhz = int(math.ceil(frequencymhz / 10.0))
    ↪ *10 #Runden auf volle Zehner
window.Element("_output_").Update(frequencymhz) #
    ↪ Das Fenster wird mit aktueller Motorfrequenz
    ↪ aktualisiert
if button1 is None or button1 == "Beenden": #
    ↪ Programm beenden
    break

window.Close()
```


Anhang C Schaltpläne

C.1 Steuerung und Auswertung des Sensors

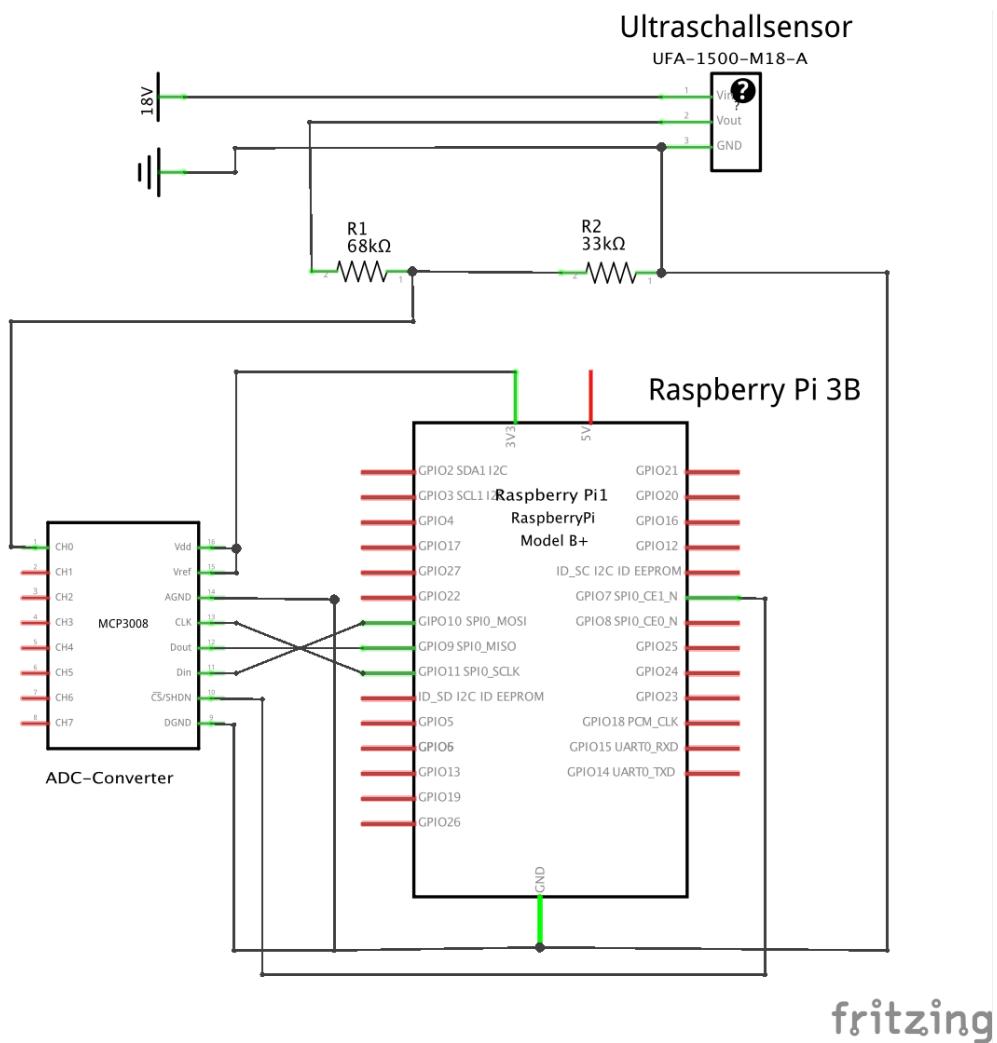


Abbildung C.1: Schaltplan der Steuerung und Auswertung des Sensors

C.2 Frequenzbestimmung mit dem Generator

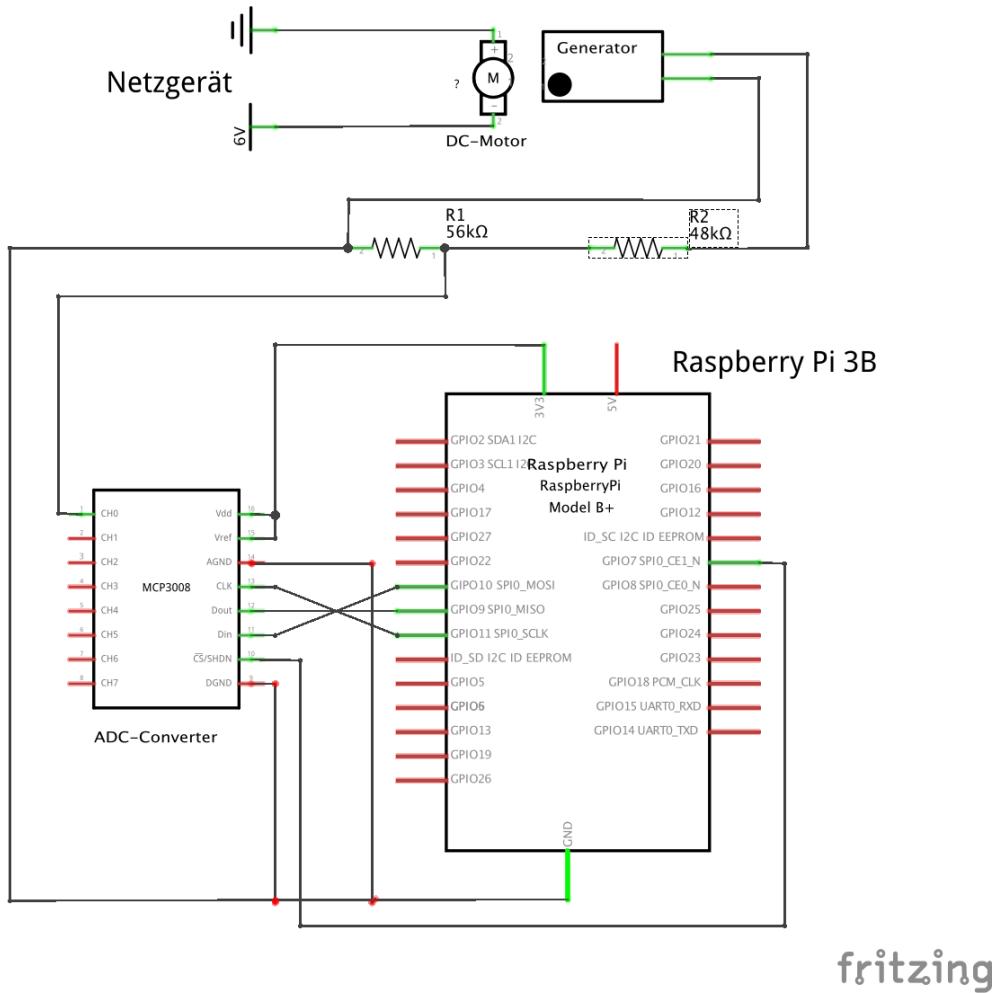
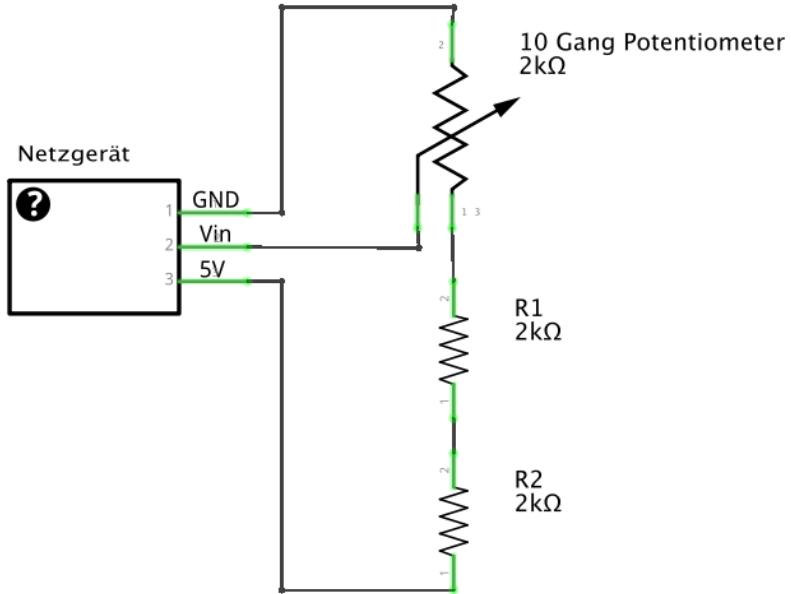


Abbildung C.2: Schaltplan der Frequenzbestimmung mit dem Generator

C.3 Remote Control Steuerung des Netzgerätes für den Motor

Remote Control Netzgerät

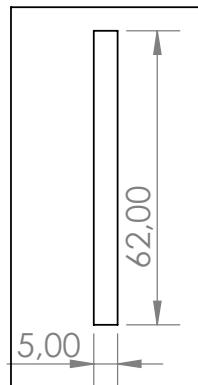
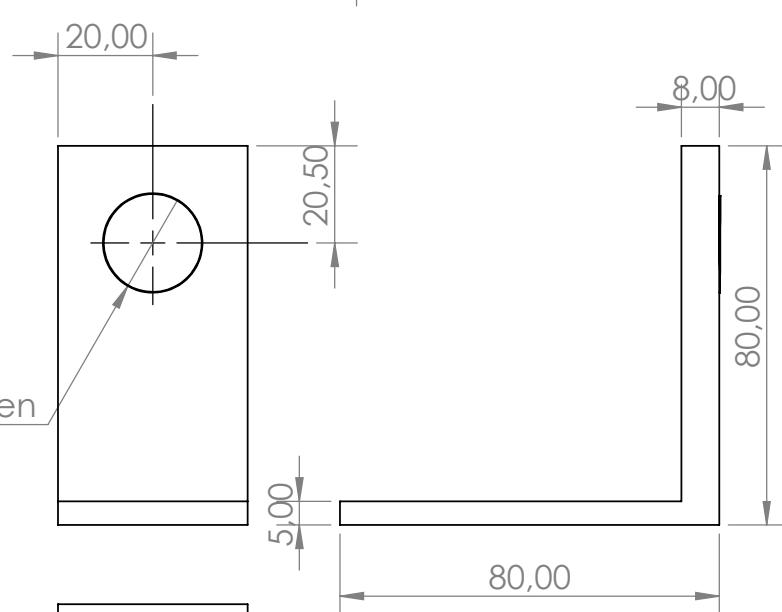
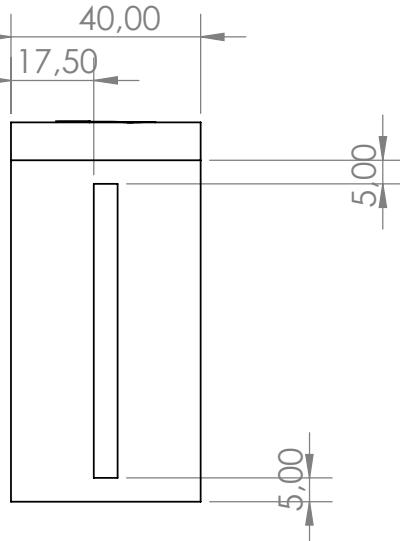
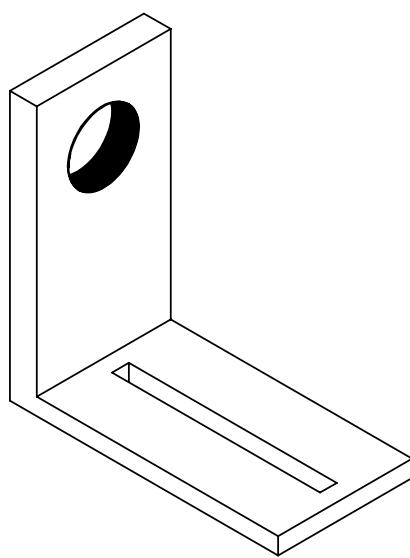


fritzing

Abbildung C.3: Schaltplan der Remote Control Steuerung des Netzgerätes für den Motor

Anhang D

Bauteile



WENN NICHT ANDERS DEFINIERT:
BEMASSUNGEN SIND IN MILLIMETER
OBERFLÄCHENBESCHAFFENHEIT:
TOLERANZEN:
LINEAR;
WINKEL:

oberflächengüte:

ENTGRATEN
UND SCHARFE
KANTEN
BRECHEN

ZEICHNUNG NICHT SKALIEREN

ÄNDERUNG

GEZEICHNET

NAME: SIGNUR: DATUM:

BENENNUNG:

GEPRÜFT

GENEHMIGT

PRODUKTION

QUALITÄT

WERKSTOFF:

ZEICHNUNGSNR.

Sensorhalterung

A4

GEWICHT:

MASSSTAB:1:2

BLATT 1 VON 1

4

3

2

1

F

E

D

C

B

A

F

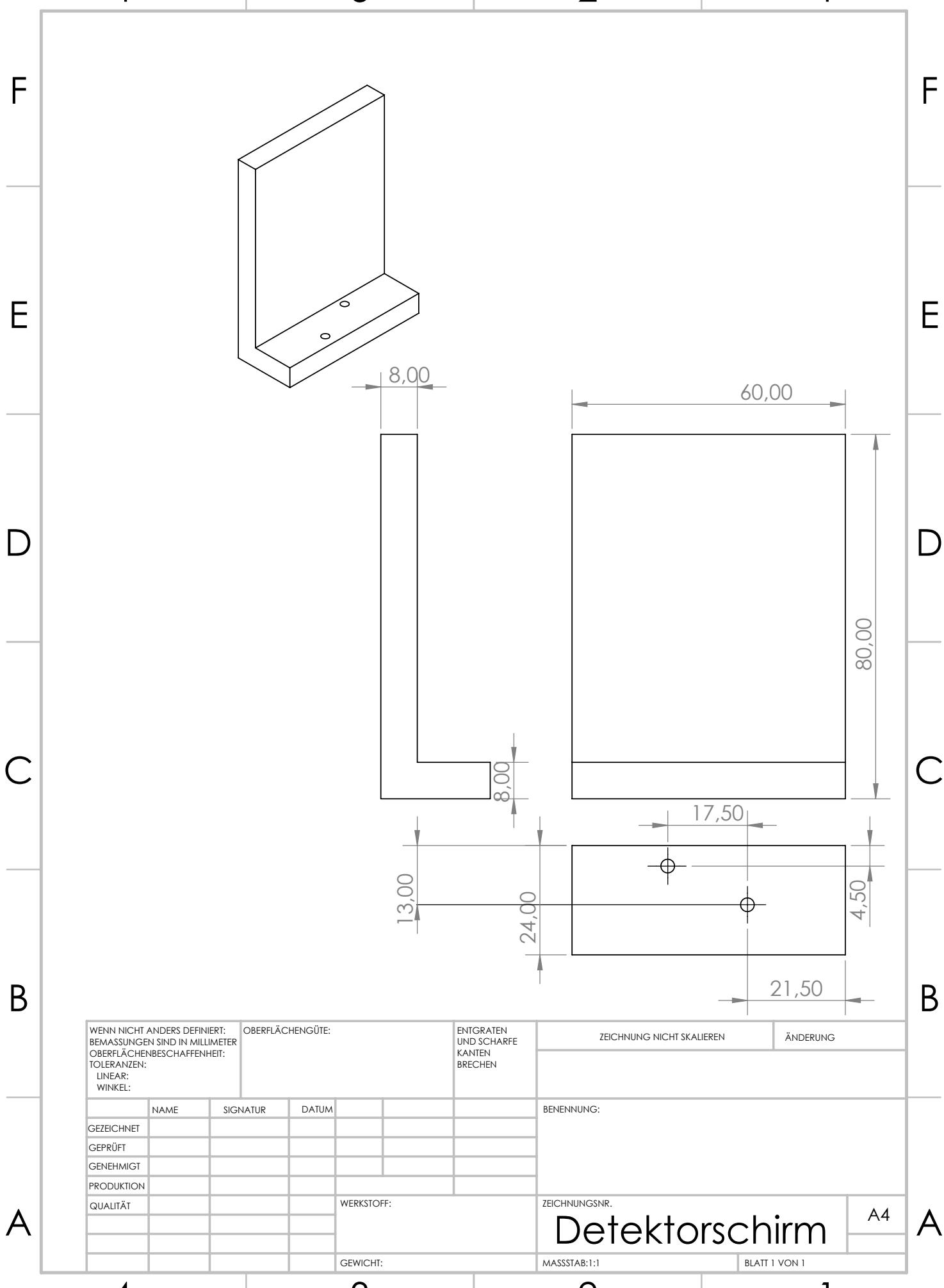
E

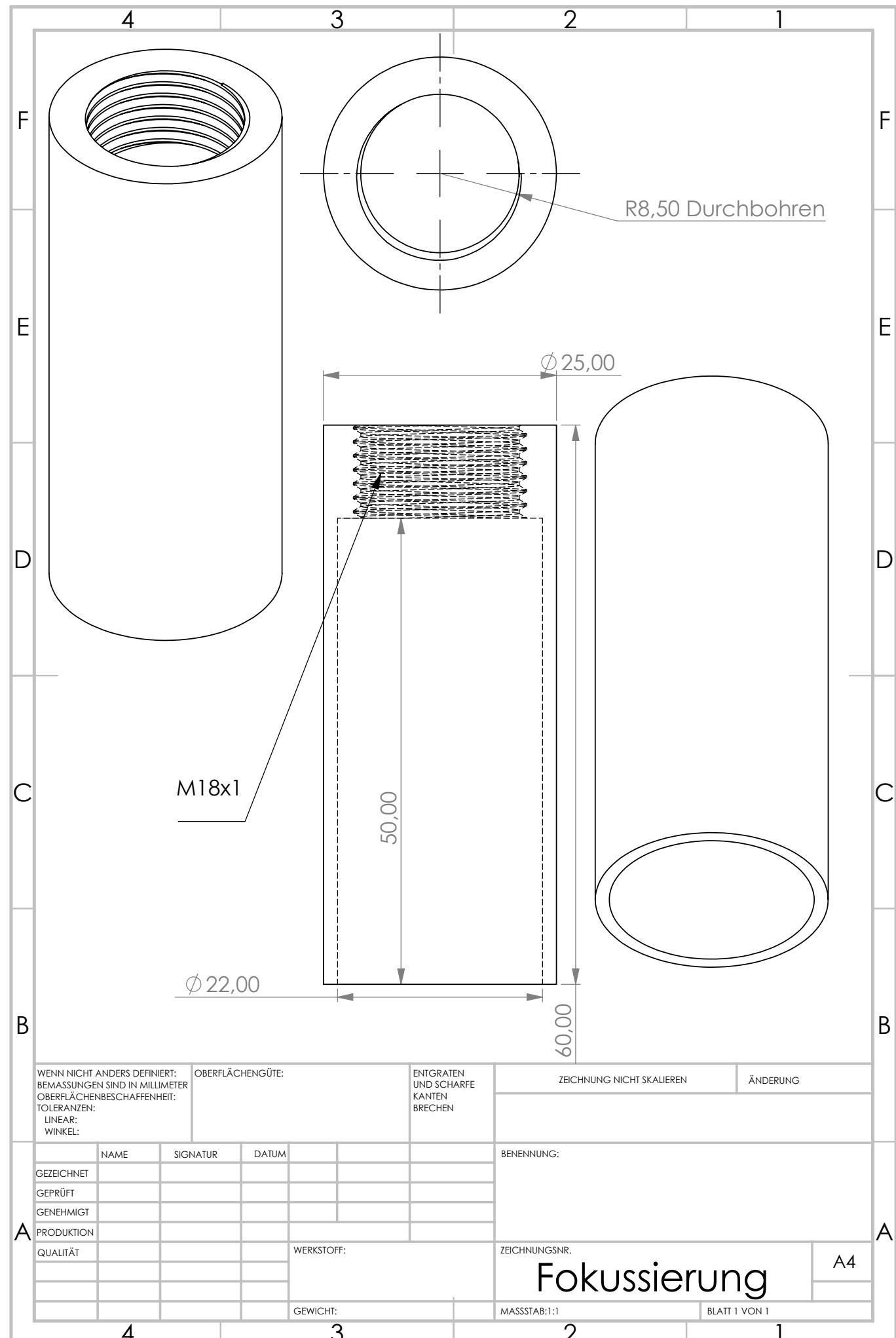
D

C

B

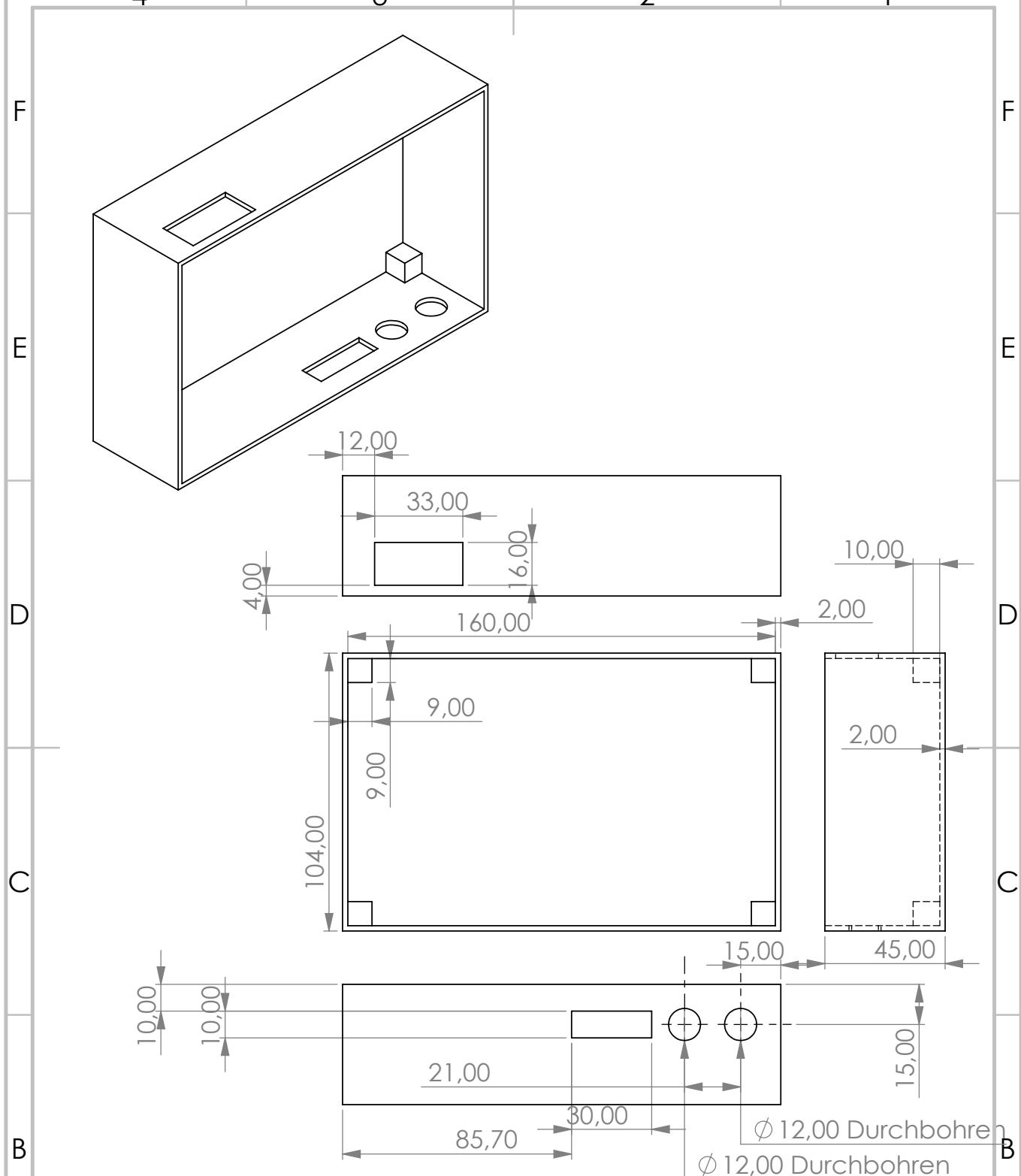
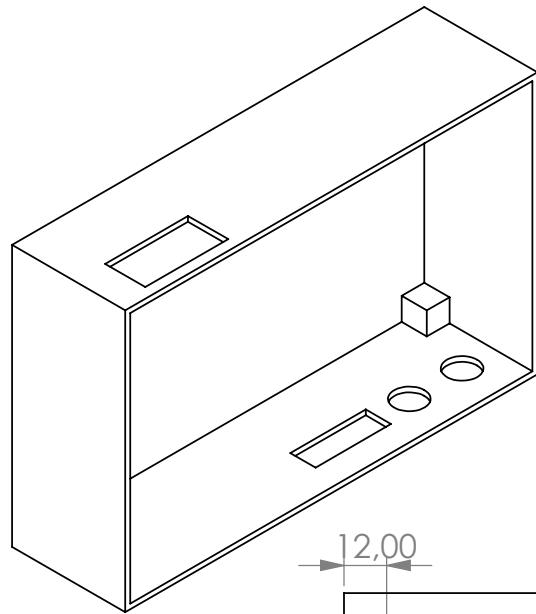
A





Anhang E

Gehäuse für den Raspberry Pi



WENN NICHT ANDERS DEFINIERT:
BEMASSUNGEN SIND IN MILLIMETER
OBERFLÄCHENBESCHAFFENHEIT:
TOLERANZEN:
LINEAR;
WINKEL:

OBERFLÄCHENGÜTE:

ENTGRATEN
UND SCHARFE
KANTEN
BRECHEN

ZEICHNUNG NICHT SKALIEREN

ÄNDERUNG

GEZEICHNET	NAME	SIGNATUR	DATUM
GEPRÜFT			
GENEHMIGT			
PRODUKTION			
QUALITÄT			

BENENNUNG:

ZEICHNUNGSNR.

Box

A4

MASSSTAB:1:5

BLATT 1 VON 1

4

3

2

1

F

E

D

C

B

A

F

E

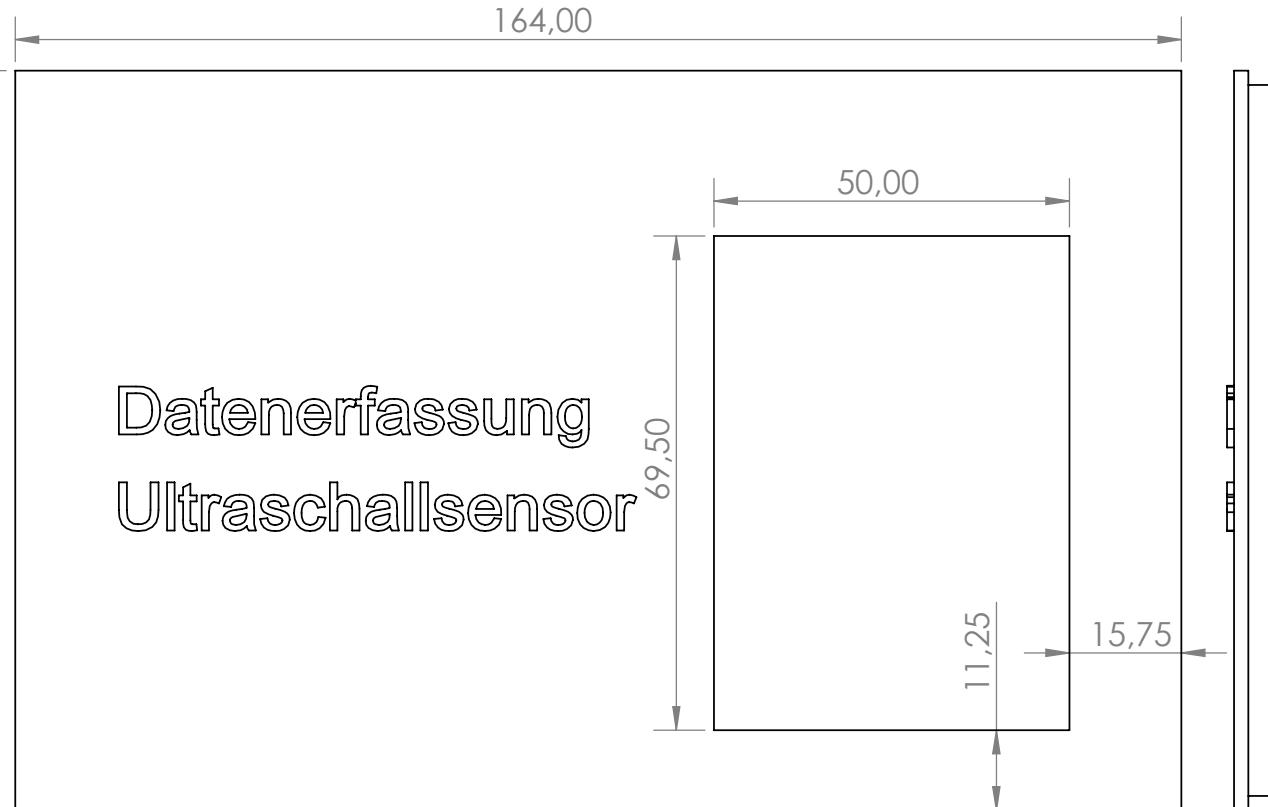
D

C

B

A

Datenerfassung Ultraschallsensor



WENN NICHT ANDERS DEFINIERT:
BEMASSUNGEN SIND IN MILLIMETER
OBERFLÄCHENBESCHAFFENHEIT:
TOLERANZEN:
LINEAR;
WINKEL:

OBERFLÄCHENGÜTE:

ENTGRATEN
UND SCHARFE
KANTEN
BRECHEN

ZEICHNUNG NICHT SKALIEREN

ÄNDERUNG

GEZEICHNET

NAME

SIGNUR

DATUM

GEPRÜFT

GENEHMIGT

PRODUKTION

QUALITÄT

BENENNUNG:

WERKSTOFF:

ZEICHNUNGSNR.

Deckel Sensor

A4

GEWICHT:

MASSSTAB:1:4

BLATT 1 VON 1

4

3

2

1

4

3

2

1

F

F

E

E

D

D

C

C

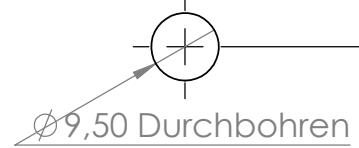
B

B

A

A

Motorsteuerung



40,00
20,00

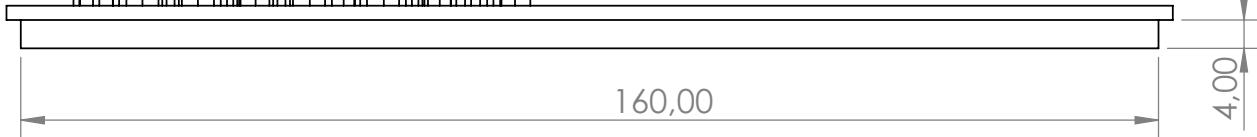
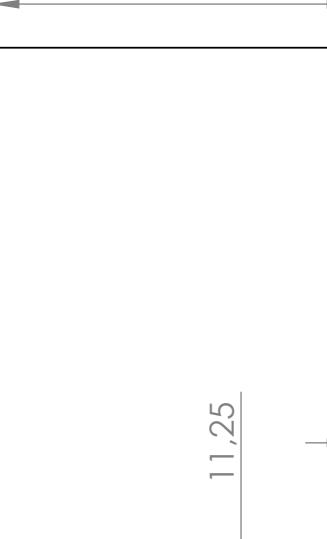
164,00

50,00

11,25
15,75

104,00

69,50



WENN NICHT ANDERS DEFINIERT:
BEMASSUNGEN SIND IN MILLIMETER
OBERFLÄCHENBESCHAFFENHEIT:
TOLERANZEN:
LINEAR;
WINKEL:

OBERFLÄCHENGÜTE:

ENTGRATEN
UND SCHARFE
KANTEN
BRECHEN

ZEICHNUNG NICHT SKALIEREN

ÄNDERUNG

GEZEICHNET

NAME

SIGNATUR

DATUM

GEPRÜFT

GENEHMIGT

PRODUKTION

QUALITÄT

BENENNUNG:

WERKSTOFF:

ZEICHNUNGSNR.

A4

Deckel Motor

MASSSTAB:1:4

BLATT 1 VON 1

4

3

2

1

Anhang F

Abbildungsverzeichnis

Abbildungsverzeichnis

1.1	Ungedämpfte Schwingung	2
1.2	Gedämpfte Schwingung	4
1.3	Erzwungene Schwingung	6
1.4	Halbbreite der Resonanzkurve	7
1.5	Versuchsaufbau[1]	8
1.6	Aufbau der Wirbelstrombremse[1]	9
2.1	Aufbau des Ultraschallsensors[5]	18
2.2	Schematische Funktionsweise eines Ultraschallsensors[6]	18
2.3	Ein- und Ausgangssignal des HC-SR04[7]	19
2.4	Schaltplan des Ultraschallsensors HC-SR04 am Raspberry Pi[7]	21
2.5	Aufbau Spannungsteiler[8]	22
2.6	Aufbau zur Bestimmung der Fokussierung des Messbereiches	23
2.7	Messdaten zum Test der Fokussierung des Messbereiches	24
2.8	Reflexion des Ultraschallimpulses	25
2.9	Ultraschallsensor mit Barrieren	26
2.10	Befestigung des Ultraschallsensors auf der optischen Bank	27
2.11	Messwerte zur Messgenauigkeit des Ultraschallsensors	27
2.12	Strahlenverlauf bei der Infrarot-Triangulation[9]	28
2.13	Skizze der Photodiode[9]	29
2.14	Ausgangskennlinie des Infrarotsensors für weißes und graues Papier[10]	30
2.15	Schaltplan des Infrarotsensors am Raspberry Pi[11]	32
2.16	Montage des Infrarotsensor auf der optischen Bank	33
2.17	Ausgangsspannung des Infrarotsensors in Abhängigkeit der Entfernung	33
2.18	Ausgangsspannung des Infrarotsensors in Abhängigkeit der Entfernung bei zwei Messreihen	34
2.19	Aufbau zur seitlichen Drehung des Spiegels	35
2.20	Winkel der maximalen Ausgangsspannung und maximale Ausgangsspannung in Abhängigkeit der Entfernung bei dem Infrarotsensor . .	36
2.21	Schaltplan der Ausgangsspannung am Raspberry Pi[11]	37
2.22	3 Messreihen zur Ausgangsspannung des zweiten Ultraschallsensors in Abhängigkeit der Entfernung	38
2.23	Linearer Fit durch die 3 Messreihen des Ultraschallsensors	39

Abbildungsverzeichnis

3.1	Waycon Ultraschallsensor [12]	41
3.2	Sensorhalterung und Detektorschirm	42
3.3	Fokussierung für den Ultraschallsensor	42
3.4	Box für die Schaltung des Sensors	44
3.5	Deckel für die Schaltung des Sensors	44
3.6	Fertiges Gehäuse für die Datenverarbeitung	45
3.7	Flussdiagramm zum Aufbau der grafischen Oberfläche	46
3.8	Messfenster der grafischen Oberfläche	47
3.9	Box für die Schaltung der Motorsteuerung	49
3.10	Deckel für die Schaltung der Motorsteuerung	49
3.11	Motorfrequenz in Abhängigkeit der digitalen Einheiten inklusive einer linearen Regression	50
3.12	Fertiges Gehäuse für die Motorsteuerung	51
C.1	Schaltplan der Steuerung und Auswertung des Sensors	80
C.2	Schaltplan der Frequenzbestimmung mit dem Generator	81
C.3	Schaltplan der Remote Control Steuerung des Netzgerätes für den Motor	82

Literatur

- [1] *Physikalisches Praktikum Weihenstephan - Anleitungen zu den Versuchen.* 29. Juli 2019.
- [2] MTS Sensor Technologie GmbH Co. KG. *Wegmessung mit Köpfchen.* URL: <https://kem.industrie.de/allgemein/wegmessung-mit-koepfchen/> (besucht am 29.07.2019).
- [3] ASM Automation Sensorik Messtechnik GmbH. *Magnetband.* URL: <https://www.asm-sensor.com/de/magnetband/magnetband.html> (besucht am 29.07.2019).
- [4] WayCon Positionsmesstechnik. *Ultraschallsensoren.* URL: <https://www.waycon.de/produkte.ultraschallsensoren/> (besucht am 29.07.2019).
- [5] Adnan Aqeel. *Introduction to HC-SR04.* URL: <https://www.theengineeringprojects.com/2018/10/introduction-to-hc-sr04-ultrasonic-sensor.html> (besucht am 29.07.2019).
- [6] Nicolas Pannwitz. *HC-SR04.* URL: <https://pic-projekte.de/blog/hr-sr04/> (besucht am 29.07.2019).
- [7] *Datasheet Ultraschall Messmodul HC-SR04.* 29. Juli 2019.
- [8] Elektronik Kompendium. *Spannungsteiler / Spannungsteilerschaltung.* URL: <https://www.elektronik-kompendium.de/sites/slt/0201111.htm> (besucht am 29.07.2019).
- [9] Moritz Ulrich Karl Aupperle. ‘Konzeption und Gestaltung eines digitalen Messwerterfassungs- systems für den Physikunterricht in der Schule’. Masterarbeit. Karlsruher Institut für Technologie KIT, 27. Nov. 2018.
- [10] *Datasheet Sharp GP2Y0A21YK0F.* 29. Juli 2019.
- [11] Raspberry Pi Tutorials - Felix. *Infrarot Abstandsmessung mit dem Raspberry Pi.* URL: <https://tutorials-raspberrypi.de/infrarot-abstandsmessung-mit-dem-raspberry-pi-sharp-gp2y0a02yk0f/> (besucht am 29.07.2019).
- [12] *Datasheet Ultraschall-Abstands- und Näherungssensoren / Serie UFA-1500.* 29. Juli 2019.
- [13] The PySimpleGUI Organization - Michael Barnett. *PySimpleGUI User’s Manual.* URL: <https://pysimplegui.readthedocs.io/en/latest/> (besucht am 29.07.2019).

- [14] Raspberry Pi Tutorials - Felix. *Am Raspberry Pi analoge Signale auslesen*. URL: <https://tutorials-raspberrypi.de/raspberry-pi-mcp3008-analoge-signale-auslesen/> (besucht am 29.07.2019).
- [15] Raspberry Pi Tutorials - Felix. *Entfernungen messen mit Ultraschallsensor HC-SR04*. URL: <https://tutorials-raspberrypi.de/entfernung-messen-mit-ultraschallsensor-hc-sr04/> (besucht am 29.07.2019).