Hochschule Konstanz
Technik, Wirtschaft und Gestaltung

Signale, Systeme und Sensoren

Kalibrierung und Einsatz eines Infrarot-Entfernungsmessers

Daniel Wollmann, Vlad Bratulescu

Konstanz, 15. November 2020

Zusammenfassung (Abstract)

Thema:	Kalibrierung und Einsatz eines Infrarot-Entfernungsmessers	
Autoren:	Daniel Wollmann	da161wol@htwg-konstanz.de
	Vlad Bratulescu	vl161bra@htwg-konstanz.de
Betreuer:	Prof. Dr. Matthias O. Franz	mfranz@htwg-konstanz.de
	Jürgen Keppler	juergen.keppler@htwg-konstanz.de
	Mert Zeybek	me431zey@htwg-konstanz.de

In diesem Versuch werden die in der Vorlesung behandelten Techniken zur Kalibrierung, Fehleranalyse und Fehlerrechnung auf den Fall eines Entfernungsmessers angewandt. Zur Berechnung der Distanz wird ein Sensor verwendet, der nach dem Triangulationsprinzip arbeitet.

Dabei wird ein Lichtstrahl ausgesendet, welcher vom Objekt reflektiert und zu einem optischen Positionssensor gelenkt wird. Die Leitfähigkeit dieses OPS ist abhängig davon, an welcher Stelle der Lichtstrahl einfällt. Sie wird mit einem Signalprozessor in eine Spannung umgewandelt, die am Ausgang des Sensors ausgegeben wird.

Die Ausgangsspannung ist anti-proportional zur steigenden Entfernung. Laut Datenblatt liegt der Messbereich des Sensors zwischen 10 cm und 80 cm.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listingverzeichnis

1

Einleitung

In diesem Versuch wird ein Distanzsensor der Firma Sharp (GP2Y0A21YK0F, siehe Datenblatt in Moodle) kalibriert.

Dabei werden 20 Messungen mit verschiedenen Entfernungen durchgeführt und die Ausgangsspannung aufgezeichnet.

Mithilfe der Software PicoScope 6 werden die Messwerte auch in digitaler Form als .csv Dateien gespeichert, welche mit einem Python Skript weiter verarbeiten werden können. Dazu gehört die Berechnung des Mittelwerts und der Standardabweichung.

Um die Ausgleichsfunktion zu bestimmen, wird das Verfahren der Linearen Regression angewendet.

Durch die Ausgleichsfunktion kann die Ausgangsspannung in die Distanz umgerechnet werden und die Fläche eines DIN-A4 Blatts anhand der gemessenen Länge und Breite berechnet werden. Um die Genauigkeit dieser Fläche zu bestimmen, wird eine Fehlerrechnung anhand der in der Vorlesung besprochenen Verfahren durchgeführt.

2

Versuch 1

2.1 Fragestellung, Messprinzip, Aufbau, Messmittel

Im Versuch 1 soll Kennlinie des Sensors mithilfe mehrer Messungen an gleichmäßigen Abständen zwischen 10 cm und 70 cm ermittelt werden. Die Messungen sollen sowohl automatisch mittels einer Software, sowie von Hand erfasst werden.

Der Distanzsensor wird mit einer Spannungsquelle von 5 V Gleichspannung versorgt. Des weiteren werden die Anschlüsse für Signalausgang und Ground mit dem Osziloskop verbunden. Das Osziloskop wird per USB-Anschluss mit dem Computer verbunden, um die Messwerte automatisch zu erfassen.

Als Messobjekt wird eine weiße Holzplatte in festgelegten Abständen vor dem Sensor platziert und die Messung durchgeführt. Es werden 20 Messungen mit verschiedenen Distanzen vorgenommen. Die Messungen werden in gleichmäßigen Abständen zwischen 10 und 70 cm durchgeführt. Zur Platzierung der Holzplatte wird ein Meterstab verwendet.

Die Erfassung der Messwerte erfolgen von Hand, sowie auch durch das Computerprogramm PicoScope 6.

Herr Keppler hat den Versuch online über ein WebEx Meeting für uns durchgeführt.

2.2 Messwerte

Tabelle [2.1] zeigt die handschriftlich aufgeschriebenen sowie per Python-Skript berechneten Spannungsmittelwerte der verschiedenen Distanzen. Zusätzlich werden auch die jeweiligen Standardabweichungen berechnet.

Bemerkung: die berechneten Mittelwerte wurden von den bereitgestellten Daten aus

Nr.	Entfernung [cm]	Spannung [V]	Spannung .py Skript [V]	σ .py Skript [V]
1	70	0,448	0,356	0,0175
2	67	0,514	0,357	0,0175
3	65	0,525	0,396	0,0176
4	62	0,523	0,434	0,0175
5	60	0,565	0,453	0,0174
6	57	0,610	0,473	0,0173
7	55	0,627	0,496	0,0174
8	52	0,610	0,535	0,0169
9	50	0,545	0,575	0,0170
10	47	0,553	0,612	0,0168
11	43	0,608	0,632	0,0176
12	40	0,616	0,671	0,0174
13	37	0,635	0,709	0,0175
14	33	0,671	0,749	0,0177
15	30	0,713	0,807	0,0173
16	27	0,752	0,907	0,0170
17	22	0,866	0,983	0,0172
18	18	0,972	1,078	0,0172
19	14	1,139	1,196	0,0173
20	10	1,353	1,341	0,0217

Tabelle 2.1: Messwerte Kalibrierung

Moodle berechnet und stehen nicht in Zusammenhang mit den von Hand aufgeschriebenen Spannungen.

2.3 Auswertung

Als nächstes werden die Daten mit Python eingelesen. Mithilfe der Spannungen können dann der Mittelwert und die Standardabweichung zu den einzelnen Dateien berechnet werden. Zur Veranschaulichung werden mit einem Python Skript A.1.1 die Abbildungen 2.1 und 2.2 erstellt.

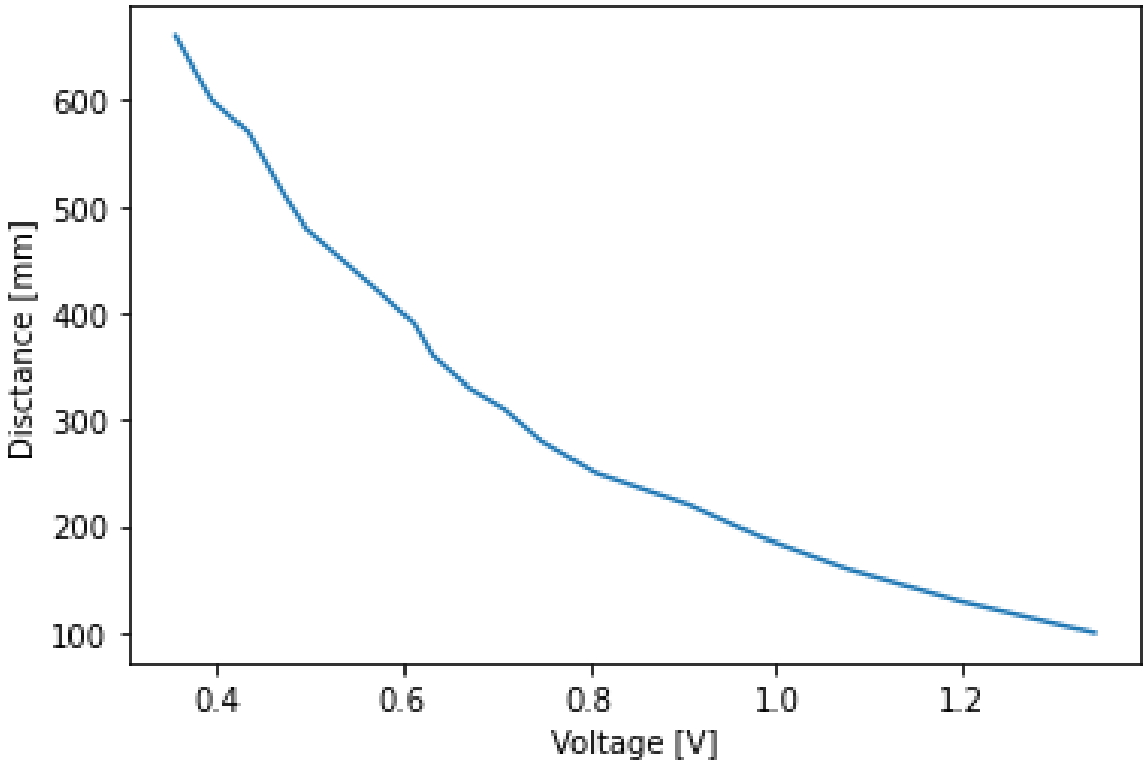


Abbildung 2.1: Spannung und Distanz

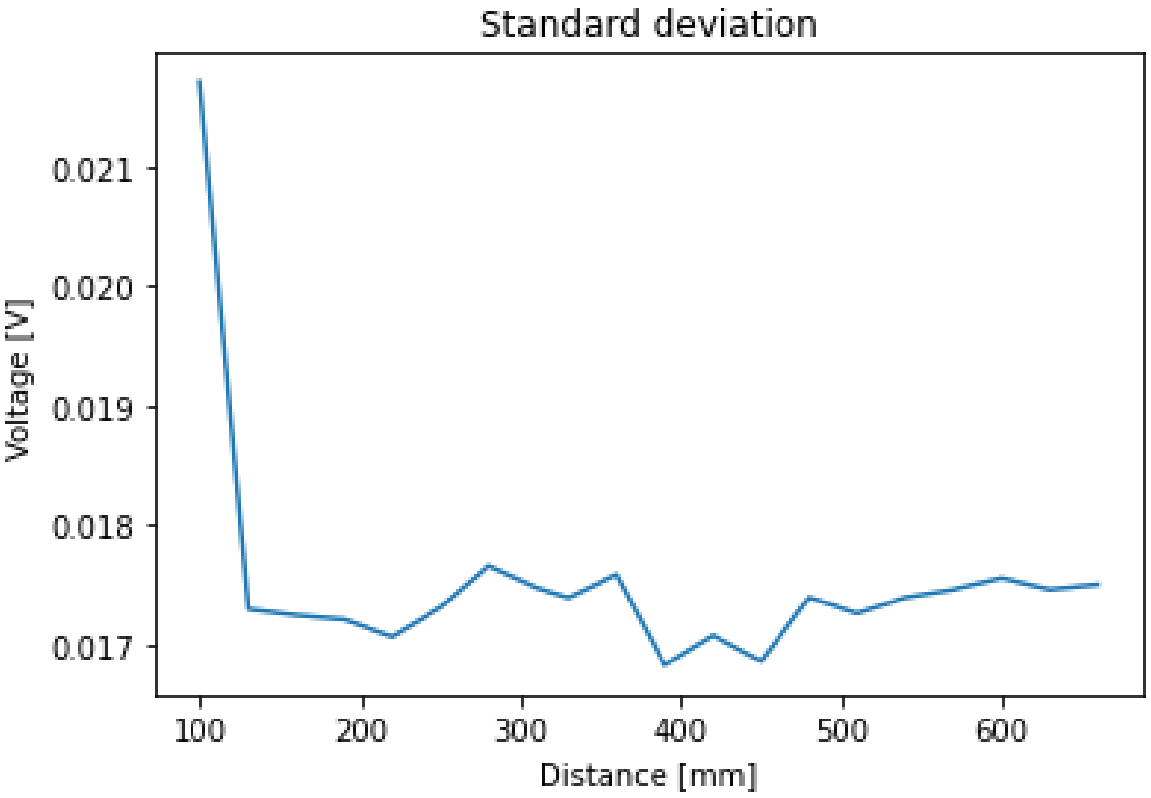


Abbildung 2.2: Standardabweichung und Distanz

2.4 Interpretation

Die Auswertung der Messwerte verdeutlicht die anti-proportional Ausgangsspannung zur Distanz.

Es ist zu erkennen, dass die von Hand aufgeschriebenen Spannungen teilweise sehr ungenau sind, da zum Beispiel die handschriftlichen Messungen 6 bis 10 ein Verhalten zeigen, dass die Anti-Proportionalität nicht widerspiegelt.

Die ermittelten Spannungen des Python Skripts zeigen ein deutlich akkurateres Ergebnis, da 1500 Einzelmessungen zur Berechnung verwendet wurden.

3

Versuch 2

3.1 Fragestellung, Messprinzip, Aufbau, Messmittel

In Versuch 2 war es die Aufgabe alle Distanzen und Mittelwerte der von Hand aufgeschriebenen Spannungen aus der Tabelle zu logarithmieren und den Zusammenhang graphisch darzustellen. Hierbei soll die resultierende Kennlinie die Form einer Geraden haben.

Anschließend wird mithilfe der linearen Regression die Ausgleichsgerade in Python berechnet.

Allerdings lässt sich dieses Verfahren nur bei einer linearen Kennlinie anwenden, welcher bei unserem Sensor nicht vorliegt. Um das Verfahren trotzdem nutzen zu können, wurde der im Aufgabenblatt beschriebene Lösungsansatz verwendet.

Für den Versuch werden lediglich die manuell erfassten Daten in der Tabelle [2.1] benötigt und ein Python Skript, das mittels der Formeln und den Daten die Berechnung durchführt.

3.2 Messwerte

Tabelle [3.1] zeigt die in Python logarithmierten Werte.

Entfernung [cm]	**log mean**	**log distance**
70	-0,80296	6,55108
67	-0,66553	6,50728
65	-0,64436	6,47697
62	-0,64817	6,42972
60	-0,57093	6,39693
57	-0,49430	6,34564
55	-0,46681	6,30992
52	-0,49430	6,25383
50	-0,60697	6,21461
47	-0,59240	6,15273
43	-0,49758	6,06379
40	-0,48451	5,99146
37	-0,45413	5,91350
33	-0,39899	5,79909
30	-0,33827	5,70378
27	-0,28502	5,59842
22	-0,14387	5,39363
18	-0,02840	5,19296
14	0,13015	4,94164
10	0,30232	4,60517

Tabelle 3.1: Logarithmierte Messwerte

3.3 Auswertung

Die manuell erfassten Spannungen und Distanzen werden jeweils mithilfe eines Python Skripts A.1.2 logarithmiert. Deren Zusammenhang wird dann in Abbildung 3.1 dargestellt.

Mithilfe der linearen Regression, soll dazu auch die Ausgleichsgerade berechnet werden. Die in der Vorlesung vorgestellte Methode der linearen Regression, funktioniert für den Sharp-Sensor nicht, da er die nichtlineare Kennlinie der Form

$$y = x^a$$

besitzt. Aus diesem Grund wird auf die obere Gleichung die doppelte Logarithmierung angewendet, sodass sich der folgende Zusammenhang daraus ergibt:

$$y' = a \cdot x' + b$$

wobei x' hier die logarithmierte Spannung und y' der logarithmierte Abstand darstellt. Die Parameter dieser Gerade lassen sich dann durch die lineare Regression schätzen.

$$a = -1.915981$$

$$b = 5,157991$$

Abbildung 3.1 stellt diese Ausgleichungsgerade dar.

Die Rückrechnung auf den ursprünglichen Zusammenhang geschieht über die Umkehrung der doppelten Logarithmierung:

$$y = exp(a \cdot lnx + b) = e^{5,157991} + x^{-1,915981}$$

wobei x hier die Spannungsmessung und y die daraus resultierende Abstandsmessung darstellt. In Abbildung 3.2 wird die nichtlineare Kennlinie des Sensors gezeigt.

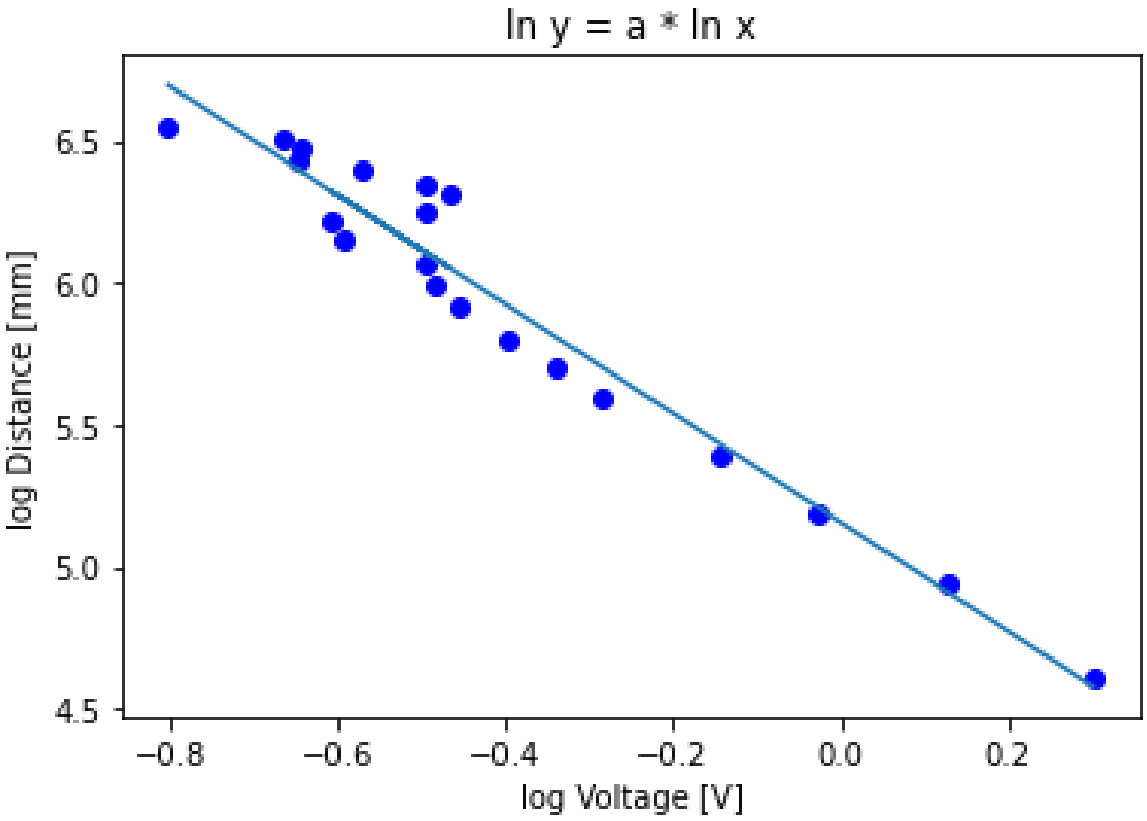


Abbildung 3.1: Lineare Regression

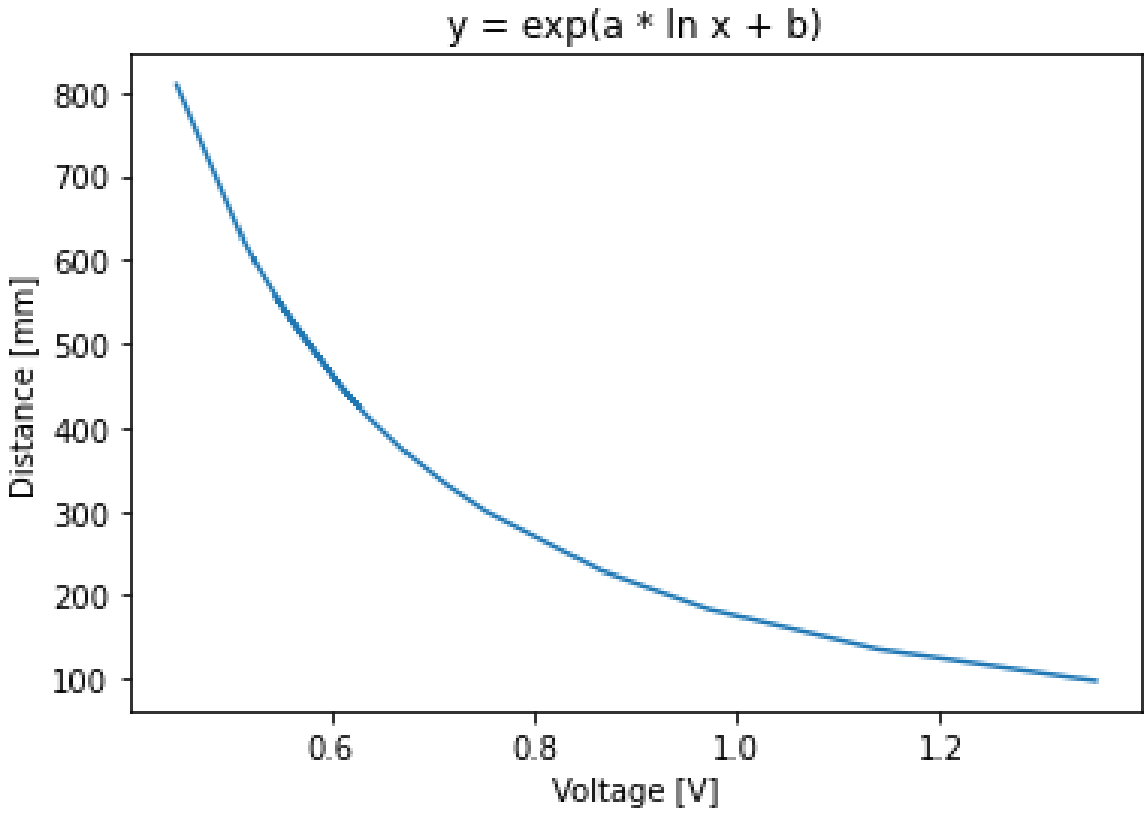


Abbildung 3.2: Kennlinie des Sensors

3.4 Interpretation

In Abbildung 3.1 kann man im Bereich von 6.25 mm Unstimmigkeiten erkennen. Der Spannungswert steigt in diesem Bereich und fällt dann wieder zurück auf die Kennlinie der Gerade. Da wegen der aktuellen Situation die Messung nicht von uns selber gemacht wurde, können wir die Ursache nicht genau bestimmen. Es ist anzunehmen, dass die Messungen dieser Abstände nicht korrekt durchgeführt wurden.

Abbildung 3.1 zeigt die Ausgleichsgerade der einzelnen Punkte. Diese scheint minimal beeinflusst von den einzelnen Ausreißern. Die andere Punkte liegen sehr nah an der Geraden.

Zum Schluss wird in Abbildung 3.2 die nichtlineare Kennlinie des Sensors berechnet. Diese zeigt, dass die Parameter a und b trotz des beschriebenen Messfehlers gut geschätzt werden konnten.

4

Versuch 3

4.1 Fragestellung, Messprinzip, Aufbau, Messmittel

Der dritte Versuch beschäftigt sich mit der Ermittlung des Messfehlers des Abstandssensors. Zu Beginn soll die lange Seite eines DIN-A4 Blattes gemessen werden und die Daten in einer .csv Datei gesichert werden. Zur Ermittlung des Messfehlers muss die Fehlerfortpflanzung durch die Kennlinie $e^b \cdot x^a$ (siehe Abbildung 3.2) berechnet werden.

Des Weiteren soll der Vertrauensbereich für eine Sicherheit von 68% und 95% berechnet werden. Anschließend kann mithilfe der Fehlerfortpflanzung das Ergebnis der Abstandsmessung in korrekter Form in cm angegeben werden.

Der Zweite Teil des Versuches beschäftigt sich mit der Breite eines DIN-A4 Blattes und der gleichen Rechenmethode wie im ersten Teil des Versuches. Durch die ermittelte Breite und Länge des DIN-A4 Blattes kann nun der Flächeninhalt berechnet werden. Für die Flächenberechnung wird das Gaußsche Fehlerfortpflanzungsgesetz aus der Vorlesung angewandt.

4.2 Messwerte

In der Tabelle [4.1] sind die handschriftlich notierten Spannungen zu der Länge, wie auch der Breite des gemessenen DIN-A4 Blattes zu sehen.

Entfernung [cm]	**Durchschnittliche Spannung [V]**
29,7	0,693
21	0,890

Tabelle 4.1: Handschriftlich notierte Spannung zur Länge und Breite

4.3 Auswertung

Die Tabelle [4.2] zeigt die berechneten Messfehler im Vertrauensbereich von 68% und 95% für die lange Seite des DIN-A4 Blattes mit 29,7 cm.

Vertrauensbereich	**Spannung von [V]**	**Spannung bis [V]**
68%	0.675296	0.710704
95%	0.658300	0.727700

Tabelle 4.2: Messfehler in Gaußverteilung für die Länge

Für die Berechnung des Vertrauensbereichs wurde die Formel $x = \bar{x} \pm t \cdot \sigma$ verwendet. Wobei t der Korrekturfaktor ist und bei 68% den Faktor 1 hat und bei 95% bei 1,96 liegt.

Die Schätzung des Messfehlers wurde ebenfalls mit der selben Methode für die Breite des Blattes durchgeführt. Das Ergebnis für die Breite mit 21 cm sehen wir in Tabelle [4.3]

Vertrauensbereich	**Spannung von [V]**	**Spannung bis [V]**
68%	0.872955	0.907045
95%	0.856592	0.923408

Tabelle 4.3: Messfehler in Gaußverteilung für die Breite

Um das Ergebnis in Korrekter Form anzeigen zu können, mittels der Fehlerfortpflanzung, muss zuvor Δx für jeweils die Länge und Breite berechnet werden.

Die Berechnung ergibt sich aus folgender Formel: $\Delta x = \frac{\sigma}{\sqrt{n}}$. Wobei n die Anzahl der Messungen ist. Für das Vertrauensintervall von 95% muss die Standardabweichung noch mit dem Korrekturfaktor von 1,96 multipliziert werden.

Vertrauensbereich	Errechnete Entfernung [cm]	Delta +- [cm]
68%	21.729754	0.020588
68%	35.094505	0.044353
95%	21.729754	0.040352
95%	35.094505	0.086933

Tabelle 4.4: Abstandsmessung mit Fehlerfortpflanzung

Zusätzlich muss die Funktion $e^b \cdot x^a$ abgeleitet und mit dem errechneten Delta Wert multipliziert werden. Die Ableitung der Funktion sieht wie folgt aus: $a \cdot e^b \cdot x^{a-1}$

Jetzt kann alles eingesetzt werden und die Länge und Breite für die zwei Vertrauensbereiche errechnet werden.

In der nachfolgenden Tabelle [4.4] sind die Ergebnisse veranschaulicht.

$$\Delta A = \sqrt{(length \cdot \Delta width)^2 + (width \cdot \Delta length)^2}$$

Um den Flächeninhalt zu berechnen muss man die Länge und Breite miteinander multiplizieren und erhält somit 762.59 cm². Um den Messfehler im Vertrauensbereich von 68% angeben zu können muss man mit einer Toleranz von 1.204539 cm² (siehe Formel für ΔA) rechnen. Die Toleranz des Vertrauensbereichs von 95% liegt bei 2.360897 cm².

4.4 Interpretation

Leider bekommen wir für die Länge einen sehr ungenauen Wert heraus, da die handschriftlich aufgeschriebenen Daten sehr ungenau erfasst wurden und auch nicht zu den Messwerten der .csv Dateien passen. Da in unserem handschriftlichen Protkoll keine Standardabweichung angegeben ist, haben wir diese aus den .csv Dateien berechnet, was sich auch auf die weiteren Berechnungen auswirkt, besonders die Berechnung des Vertrauensbereichs. Dadurch verfälscht sich natürlich auch die Berechnung des Flächeninhaltes immens. Desweiteren ist zu erkennen, dass das berechnete Delta bei 95% fast genau doppelt so hoch ist wie von 68%. Die Beobachtung lässt sich auf die Multiplizierung des Korrekturfaktors zurückführen.

Anhang

A.1 Quellcode

A.1.1 Quellcode Versuch 1

```
import numpy as np
import matplotlib.pyplot as p;
import glob
from natsort import natsorted

# AUFGABE 1

# Dateien werden geladen und nach dem Namen sortiert.
allFiles = glob.glob("/Users/vladb/Git/htwg/S3/SSS/Versuch 1/messwerte/*.csv")
allFiles = natsorted(allFiles)

# Die Arrays fuer die Spannungen, Standardabweichung und Distanzen werden erstellt
arrayMean = np.zeros(20)
arrayStd = np.zeros(20)
arrayDistance = [100,130,160,190,220,250,280,310,330,360,390,420,450,480,510,540,570,600,630,660]

# Index fuer Arrays Traversierung
index = 0

print("Mittelwert und Standardabweichung der einzelnen Dateien")

# Es werden alle Dateien jeweils verarbeitet
for filePath in allFiles:
    # Spalten trennen. Die Werte der DSpannungen befinden sich in der fuenften Spalte
    # Die ersten 1000 Werte werden weggelassen.
    inputData = np.genfromtxt(filePath, delimiter = ',')
    voltage = inputData[1000:, 4]

```

```
    # Hier wird die der Spannungsmittelwert und Standardabweichung anhand der Stichproben berechnet
    arrayStd[index] = np.std(voltage)
    arrayMean[index] = voltage.mean()
    print("Datei: m%2d, Mittelwert: %.3f, Standardabweichung: %.5f" % (index+1,
                                                arrayMean[index],
                                                arrayStd[index]))
    index = index + 1

print("")

# Zusammenhang zwischen Durchschnittsspannung und Distanz
p.plot(arrayMean, arrayDistance)
p.ylabel('Distance [mm]')
p.xlabel('Voltage [V]')
p.show()

# Zusammenhand zwischen Distanz und Standardabweichung
p.plot(arrayDistance, arrayStd)
p.title("Standard deviation")
p.ylabel("Voltage [V]")
p.xlabel("Distance [mm]")
p.show()
```

Listing 5.1: Skript Versuch 1

A.1.2 Quellcode Versuch 2

```
import numpy as np
import matplotlib.pyplot as p;

# AUFGABE 2

# Manuell erfassten Spannungswerte
arrayMean = [1.353, 1.139, 0.972, 0.866, 0.752, 0.713, 0.671, 0.635, 0.616, 0.608, 0.553,
        0.545, 0.610, 0.627, 0.610, 0.565, 0.523, 0.525, 0.514, 0.448]
arrayDistance = [100, 140, 180, 220, 270, 300, 330, 370, 400, 430, 470, 500, 520, 550,
            570, 600, 620, 650, 670, 700]

# Alle Spannungen und Distanzen logarithmieren
logVoltage = np.log(arrayMean)
logDistance = np.log(arrayDistance)

# Berechnung der Mittelwerte
meanLogVoltage = logVoltage.mean()
meanLogDistance = logDistance.mean()

# Zusammenhang zwischen logarithmierte Distanz und logarithmiere Spannung
p.plot(logVoltage, logDistance)
p.ylabel('log Distance [mm]')
p.xlabel('log Voltage [V]')
p.show()

# Einzelne Paare ausgeben. Werden für die Erstellung der Tabelle benötigt
for i in range(len(logVoltage)):
    print("Log voltage: %.5f, Log dist: %.5f" % (logVoltage[i], logDistance[i]))

a1 = 0
a2 = 0

# Berechnung von a1 und a2 für lineare Regression
for i in range(len(logVoltage)):
    a1 += (logVoltage[i] - meanLogVoltage) * (logDistance[i] - meanLogDistance)
    a2 += pow((logVoltage[i] - meanLogVoltage), 2)

# Berechnung der a und b Parameter
a = a1 / a2
b = meanLogDistance - a * meanLogVoltage
```

```
41
42 # Ausgabe der Parameter
43 print("Variablen für Lineare Regression")
44 print("a: %f" % (a))
45 print("b: %f" % (b))
46 print("")
47
48 # Abbildung Ausgleichsgerade y'= a*x' + b
49 p.plot(logVoltage, logDistance, 'bo')
50 p.plot(logVoltage, [a * x + b for x in logVoltage])
51 p.title("ln y = a * ln x")
52 p.ylabel("log Distance [mm]")
53 p.xlabel("log Voltage [V]")
54 p.show()
55
56 # Abbildung y = exp(a * lnx + b)
57 p.plot(arrayMean, [np.exp(a * x + b) for x in logVoltage])
58 p.title("y = exp(a * ln x + b)")
59 p.ylabel("Distance [mm]")
60 p.xlabel("Voltage [V]")
61 p.show()
```

Listing 5.2: Skript Versuch 2

A.1.3 Quellcode Versuch 3

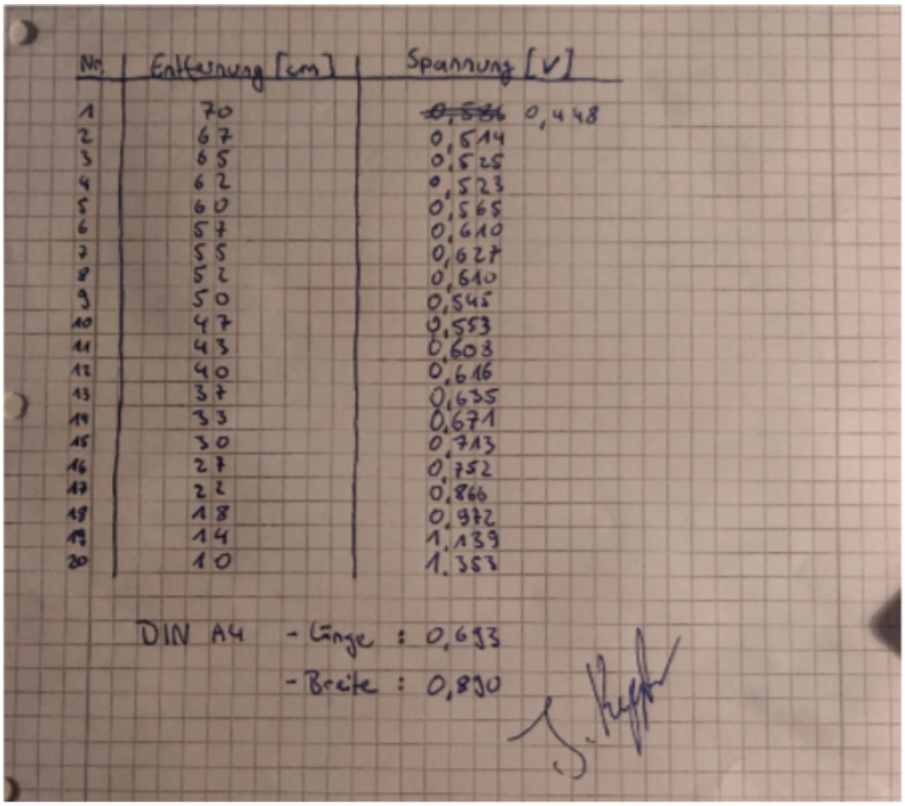
```
import numpy as np
import math
# Flächenmessung AUFGABE 3

#voltageWidthData = np.genfromtxt("/Users/vladb/Git/htwg/S3/SSS/Versuch 1/messwerte/dina4b.csv" , delimiter= ",")
#voltageLengthData = np.genfromtxt("/Users/vladb/Git/htwg/S3/SSS/Versuch 1/messwerte/dina4l.csv" , delimiter= ",")

#voltageWidth = voltageWidthData[1000:, 4]
#voltageLength = voltageLengthData[1000:, 4]

fileWidth = open("/Users/vladb/Git/htwg/S3/SSS/Versuch 1/messwerte/Blatt Messungen/dina4b.csv")
fileLength = open("/Users/vladb/Git/htwg/S3/SSS/Versuch 1/messwerte/Blatt Messungen/dina4l.csv")

voltageWidth = np.genfromtxt(fileWidth , delimiter= ",", skip_header=1000, usecols = (4))
voltageLength = np.genfromtxt(fileLength , delimiter= ",", skip_header=1000, usecols = (4))

# STD nicht im Protokol deshalb von Messungen berechnet!
# Width
sizeWidth = voltageWidth.shape[0]
meanWidth = 0.890 #von handschriftlichem Aufschrieb
stdWidth = np.std(voltageWidth)

interval68W = [meanWidth − stdWidth, meanWidth + stdWidth]
interval95W = [meanWidth − 1.96 ∗ stdWidth, meanWidth + 1.96 ∗ stdWidth]

# Length
sizeLength = voltageLength.shape[0]
meanLength = 0.693 #von handschriftlichem Aufschrieb
stdLength = np.std(voltageLength)

interval68L = [meanLength − stdLength, meanLength + stdLength]
interval95L = [meanLength − 1.96 ∗ stdLength, meanLength + 1.96 ∗ stdLength]

print("Correction and intervals:")
print("WIDTH")
print("68%% Interval: %f to %f" % (interval68W[0], interval68W[1]))
print("95%% Interval: %f to %f" % (interval95W[0], interval95W[1]))
print("")
print("LENGTH")
print("68%% Interval: %f to %f" % (interval68L[0], interval68L[1]))
```

```
41 print("95%% Interval: %f to %f" % (interval95L[0], interval95L[1]))
42 print("")
43
44 ##############################################################################
45
46 #Wird für die Fehlerfortpflanzung benötigt
47 delta68W = stdWidth / math.sqrt(sizeWidth)
48 delta68L = stdLength / math.sqrt(sizeLength)
49
50 print("WIDTH")
51 print("delta 68 u = (%f +– %f)" % (meanWidth, delta68W))
52 print("LENGTH")
53 print("delta 68 u = (%f +– %f)" % (meanLength, delta68L))
54 print("")
55
56 delta95W = stdWidth * 1.96 / math.sqrt(sizeWidth)
57 delta95L = stdLength * 1.96 / math.sqrt(sizeLength)
58
59 print("WIDTH")
60 print("delta 95 u = (%f +– %f)" % (meanWidth, delta95W))
61 print("LENGTH")
62 print("delta 95 u = (%f +– %f)" % (meanLength, delta95L))
63 print("")
64
65 ##############################################################################
66
67 #Aus Versuch 2
68 a = –1.915981
69 b = 5.157991
70
71 #Ableitung der Funktion => a*e^b*x^(a–1)
72
73 #Berechnung für Vertrauensbereich von 68%
74 width = np.exp(a * np.log(meanWidth) + b)
75 deltaWidth68 = a * np.exp(b) * math.pow(meanWidth, a – 1) * delta68W
76 deltaWidth68 = abs(deltaWidth68)
77
78 length = np.exp(a * np.log(meanLength) + b)
79 deltaLength68 = a * np.exp(b) * math.pow(meanLength, a – 1) * delta68L
80 deltaLength68 = abs(deltaLength68)
81
82 print("Correction:")
```

```
83 print("w = (%f +- %f) cm" % (width / 10, deltaWidth68 / 10))
84 print("l = (%f +- %f) cm" % (length / 10, deltaLength68 / 10))
85
86 deltaArea = math.sqrt(math.pow(length * deltaWidth68, 2) + math.pow(width * deltaLength68, 2))
87 print("area = (%f +- %f) cm^2" % (width * length / 100, deltaArea / 100))
88
89 ##############################################################################
90
91 #Berechnung für Vertrauensbereich von 95%
92 width = np.exp(a * np.log(meanWidth) + b)
93 deltaWidth95 = a * np.exp(b) * math.pow(meanWidth, a - 1) * delta95W
94 deltaWidth95 = abs(deltaWidth95)
95
96 length = np.exp(a * np.log(meanLength) + b)
97 deltaLength95 = a * np.exp(b) * math.pow(meanLength, a - 1) * delta95L
98 deltaLength95 = abs(deltaLength95)
99
100 print("Correction:")
101 print("w = (%f +- %f) cm" % (width / 10, deltaWidth95 / 10))
102 print("l = (%f +- %f) cm" % (length / 10, deltaLength95 / 10))
103
104 deltaArea = math.sqrt(math.pow(length * deltaWidth95, 2) + math.pow(width * deltaLength95, 2))
105 print("area = (%f +- %f) cm^2" % (width * length / 100, deltaArea / 100))
```

Listing 5.3: Skript Versuch 3

A.2 Messergebnisse



Nr.	Entfernung [cm]	Spannung [V]	
1	70	~~0,586~~	0,448
2	67	0,514	
3	65	0,526	
4	62	0,523	
5	60	0,565	
6	57	0,610	
7	55	0,627	
8	52	0,610	
9	50	0,545	
10	47	0,553	
11	43	0,603	
12	40	0,616	
13	37	0,635	
14	33	0,671	
15	30	0,713	
16	27	0,752	
17	22	0,866	
18	18	0,972	
19	14	1,139	
20	10	1,353	

DIN A4 - Länge : 0,693

- Breite : 0,890