Modularbeit zum HÜ-Kurs „Messen und Signalanalyse mit MATLAB®“

Thema: Kalibrierung einer Kaffeewaage von JoeFrex

Wintersemester 22/23

HM

Name, Vorname:	Viebke, Patricia
Matrikelnummer:	06433718
E-Mail:	pviebke@hm.edu
Studiengang:	Master of Applied Research in Engineering Sciences
Hochschule:	Hochschule für angewandte Wissenschaften München

Seminarleiter:	Dipl.-Ing. Armin Rohnen LfbA
Abgabedatum:	29.12.2022

HM

Inhaltsverzeichnis

Abkürzungen

g	Gramm
GPIO	General Pin Input Output
CSV	Comma Separated Values
SSH	Secure Shell
WLAN	Wireless LocalArea Network

Abstract

In dieser Modularbeit wird die Kalibrierung einer Kaffeewaage von JoeFrex schrittweise erläutert. Die Waage wird in einem Messsystem eingesetzt, dass unter anderem den Kaffeeausfluss misst. Hierfür ist eine Kalibrierung der Waage bzw. der verbauten Wägezelle notwendig. Für den Kalibrierprozess werden Eichgewichte bis 800 Gramm verwendet, damit der gesamte Messbereich abgedeckt wird. Das Messsignal der Waage wird über ein Raspberry Pi gemessen. Zwischen Waage und Raspberry befindet sich noch eine Signalaufbereitung, die das analoge Signal ins Digitale wandelt. Mit einem Python Code auf dem Raspberry Pi wird die Messung konfiguriert. Über MATLAB wird der Kalibrierprozess gestartet und anschließend statistisch ausgewertet.

1 Aufgabenstellung

Für die Entwicklung eines Messsystems für Siebträger Espressomaschinen werden Messgeräte und Sensoren benötigt. Damit diese auch zuverlässig messen, müssen die Sensoren einer Kalibrierung unterzogen werden. Mit dem Messsystem werden Parameter wie Temperatur, Druck, Durchflussrate und Kaffeeausflussmenge aufgenommen. Für das Messen der Kaffeeausflussmenge ist der Einsatz einer Waage der Firma JoeFrex geplant.

Bevor die Waage im Messsystem eingesetzt werden kann, muss eine Kalibrierung der verbauten Wägezelle über den gesamten Messbereich von 0 bis 800 Gramm durchgeführt werden. Die Aufnahme und Weiterverarbeitung der Messwerte wird mit MATLAB durchgeführt. Ziel ist es, aus dem Kalibrierprozess eine Kennlinie zu ermitteln, die zukünftig zur Umwandlung der Messwerte in physikalische Werte der Einheit Gramm dient.

2 Theorie der Wägetechnik

Die Waage misst mit einer Wägezelle nach dem Prinzip der Dehnungsmessstreifen.

Wird Gewicht auf die Wägezelle hinzugegeben, so erfahren die Messstreifen eine positive Längenänderung. Durch die Längenänderung verändert sich folglich auch der Querschnitt der Messstreifen. Mit zunehmender Länge wird der Querschnitt der Messstreifen kleiner. Durch die Verringerung des Querschnittes erhöht sich wiederum der elektrische Widerstand. Durch die Brückenschaltung wird die relative Widerstandsänderung ausgewertet [1]. Die verwendete Wägezelle ist mit einer Vollbrücke geschalten.

3 Messaufbau

Die Wägezelle der verwendeten Waage von Joe Frex muss mit der Messelektronik verbunden werden. Die eingebaute Wägezelle der Waage ist mit einer Vollbrücke geschalten, das bedeutet sie hat vier Litzen, die aus der Wägezelle herausführen. Die vier Litzen sind sehr kurz und werden daher mit einem Stecker und einem Mikrofonkabel versehen. Das Mikrofonkabel wird an die zusätzliche Signalaufbereitung angeschlossen.

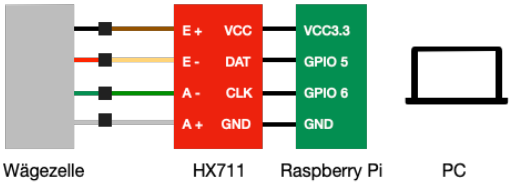

Abbildung 1: schematischer Messaufbau

Die Wägezelle ist über die vier Litzen mit der Platine für die Signalaufbereitung verbunden. Die Anschlüsse mit den Bezeichnungen $E+$ und $E-$ stehen für „excitation“. Das ist Englisch und steht für die Versorgung. Die beiden Kabel an den Anschlüssen $A+$ und $A-$ sind für die Signalübertragung zuständig, wobei $A-$ die Signalmasse darstellt. Auf der Aufbereitungsplatine befindet sich der Baustein HX711 [2]. Dieser ist für die Umwandlung des analogen Signals der Wägezelle der Einheit V/V in die digitalen 24-Bit Werte zuständig. Die ausgegebenen Werte befinden sich im Bereich zwischen 0 und 2^{24} (= 16.777.216).

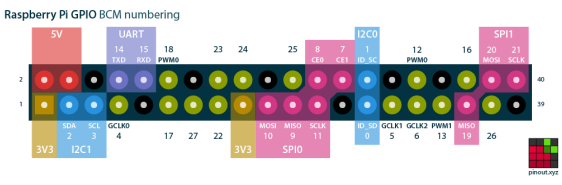

Abbildung 2: Raspberry Pi Pinout [3]

Die Ausgänge der Aufbereitungsplatine werden mit dem 40-Pin GPIO des Raspberry Pis verbunden. Die Messkette benötigt eine Stromversorgung, daher muss ein Kabel entweder an die 3.3 V oder an den 5 V Pin gelegt werden. In diesem Fall wurde der 5 V Pin verwendet. Weiter muss ein Pin auf Masse gelegt werden. Die GPIO-Pins 5 und 6 werden mit dem DAT und CLK Ausgang verbunden.

Zuletzt muss das Raspberry Pi mit Strom versorgt werden. Hierfür wird ein Micro-USB Kabel an das Raspberry Pi angeschlossen, welches über ein 5 V Netzteil Strom liefert. Für das Erhalten der Messwerte muss das Raspberry Pi und der PC an das gleiche Netzwerk gekoppelt werden. Hierfür steht beim Raspberry Pi ein Ethernet Anschluss zur Verfügung. Für den PC ist eine WLAN-Verbindung zum gleichen Netzwerk ausreichend.

Damit ist der Messaufbau vollständig. Im nächsten Schritt wird die Vorgehensweise zur Kalibrierung erläutert.

4 Vorgehensweise der Kalibrierung

Die Waage hat einen Messbereich von 0 g bis 800 g. Mit Eichgewichten bis 800 g werden jeweils in 50 g Schritten Messwerte von der Waage aufgezeichnet.

4.1 Python Skript

Für die Messung wurde auf dem Raspberry Pi ein passendes Python-Skript geschrieben, in welchem die Ausgabe der Messwerte in ein CSV-File stattfindet. Das Python File hat die Bezeichnung „hx711_test.py“ und befindet sich im Raspberry Pi Verzeichnis „/home/pi/hx711py/“.

Beim Aufrufen des Python-Skriptes muss eine Zahl mitgegeben werden. Die Zahl definiert die Länge der Messdauer in Sekunden. Im Python File wird zuerst eine Initialisierung durchgeführt, welche einige Sekunden dauert. Die Zeit der eigentlichen Messung wird jedoch erst ab dann gezählt, wenn die Messung tatsächlich startet, also nach der Initialisierung.

Während der Messung wird ein Zeitstempel erzeugt, der die aktuelle Messdauer angibt. Ist dieser Wert größer oder gleich der eingegebenen Messdauer, so wird die Messung gestoppt. Während der Messung werden die Messdaten in einem Array gespeichert.

Damit der Array nach Beenden der Messung an MATLAB übergeben werden kann, wird dieser mit der numpy Library in einen numpy-Array umgewandelt. Anschließend werden die Werte in ein CSV-file geschrieben mit der Bezeichnung „csvfile.csv“, dass sich im gleichen Verzeichnis befindet, wie hx711_test.py.

Startet man die Messung im Terminal, so werden zusätzlich die Messwerte während der Messung ausgegeben und der Array mit den gesammelten Messdaten. Abschließend wird noch ein String „csv erstellt“ ausgegeben. Dieses Vorgehen dient nur zur Überprüfung der Funktionalität.

4.2 MATLAB Skript

Damit die Messung nun auch über MATLAB ausgeführt werden kann, müssen einige Schritt durchgeführt werden. Es wurde hierfür ein MATLAB Skript geschrieben, dass wie folgt aufgebaut ist.

1. Raspberry Pi connecten
2. Messung durchführen
3. CSV-file vom Raspberry Pi holen
4. CSV-file vom Raspberry Pi löschen
5. CSV-file in MATLAB importieren
6. Datenweiterverarbeitung

Für eine erfolgreiche Verbindung des Raspberry Pis mit MATLAB muss das passende AddOn installiert sein. Das AddOn „MATLAB Support Package for Raspberry Pi Hardware“ ist essenziell für das Arbeiten des Raspberrys über MATLAB.

Da das Raspberry Pi und der PC mit dem gleichen Netzwerk verbunden sind, kann eine SSH-Verbindung aufgebaut werden.

```
system(rpi, '/home/pi/hx711py/hx711_test.py
20');
```

Mit dem Befehl *raspi* wird ein Raspberry Objekt erstellt. Die drei Inputs des Befehls stellen den Hostnamen, den Usernamen und das Passwort des Raspberrys dar. Nun sollte im Workspace eine Variable mit dem Namen *rpi* erscheinen.

Die Kalibrierung wird mit insgesamt 17 Messungen durchgeführt. Eine Messung wird ohne Gewicht auf der Waage durchgeführt und anschließend wird in 50 g Schritten bis 800 g inkrementiert. Jede Messung soll eine Messdauer von 20 Sekunden haben. Damit wird sichergestellt, ob die Waage einen Drift aufweist. Ein Drift ist eine Messwertabweichung unter konstanter Last über die Zeit.

```
system(rpi, '/home/pi/hx711py/hx711_test.py
20');
```

Mit dem Befehl *system* werden Befehle von MATLAB aus auf dem Raspberry Pi ausgeführt. Als erstes muss das Raspberry Objekt ausgewählt werden, so dass definiert ist auf welchem Gerät die Anweisung ausgeführt werden soll. Der lila markierte Teil ist eigentliche Anweisung. Die Anweisung ist genauso aufgebaut, wie wenn man das Skript direkt im Terminal auf dem Raspberry ausführt. Man erkennt den Dateipfad des Skriptes wieder. Der Unterschied hierbei ist, dass direkt nach dem Dateipfad noch der Skriptname und die Zahl 20 mitgegeben werden. Damit wird das Skript aufgerufen mit einer Messdauer von 20 Sekunden. Lässt man hinter dem Befehl das Semikolon weg, so sieht man nach Beenden der Messung die gleiche Ausgabe als String, die man im Terminal bekommen würde. Also das Array mit den Messdaten und ein „csv erstellt“.

Nachdem die Messung beendet wurde, muss das erstellte CSV-File vom Raspberry Pi auf den PC geholt werden.

```
getFile(rpi, '/home/pi/hx711py/csvfile.csv');
```

Der Befehl ist gleich aufgebaut, wie der vorherige Befehl *system*. Es muss mitgeteilt werden, welche Datei aus welchem Dateipfad zu holen ist. Wird dieser Befehl in MATLAB ausgeführt, so befindet sich das File in dem gleichen Ordner, der aktuell in MATLAB geöffnet ist. In den Standardeinstellungen hat MATLAB auf der linken Seite eine Spalte mit der Bezeichung „Current Folder“. Dort sollte nun ein CSV-File mit dem Namen csvfile.csv auftauchen.

Anschließend wird mit der *system* Anweisung das CSV-File vom Raspberry Pi gelöscht.

```
system(rpi, 'sudo rm
/home/pi/hx711py/csvfile.csv');
```

Es ist zu erkennen, dass vor dem Dateipfad noch ein *‚rm‘* geschrieben ist. Das ist in der Raspberry Umgebung der Befehl für ‚remove‘, also löschen einer Datei.

Prinzipiell wäre dieser Schritt nicht notwendig, da das CSV-File jedes Mal neu überschrieben wird, jedoch ist es sicherer das File zu löschen und beim nächsten Ausführen des Skriptes ein neues erstellen zu lassen. So wird verhindert, dass Daten übrigbleiben, bzw. nicht korrekt überschrieben werden.

Nachdem die Messdaten nun auf dem PC übertragen sind, müssen diese noch in MATLAB importiert werden. Öffnet man durch Doppelklick in MATLAB die Datei „csvfile.csv“ so öffnet sich ein weiteres Fenster, welches optimalerweise die Messwerte zeilenweise anzeigt.

In dem Fenster müssen die Importoptionen erstellt werden. Unter dem Reiter „Import Selection > Generate Script“ wird ein Skript erstellt mit den gewünschten Optionen. Dies sieht für die Kalibrierung wie folgt aus.

```
opts =
delimitedTextImportOptions("NumVariables", 1);
opts.DataLines = [1, Inf];
opts.Delimiter = ",";
opts.VariableNames = "VarName1";
opts.VariableTypes = "double";
opts.ExtraColumnsRule = "ignore";
opts.EmptyLineRule = "read";
```

Für die 17 Messungen wird immer die gleiche Anzahl an Messwerten in das CSV-File geschrieben, daher kann immer der gleiche Code verwendet werden, um das CSV-File in MATLAB zu importieren. Es wird eine Variable *opts* im Variable Workspace angelegt. Im letzten Schritt muss das CSV-File einer Variable zugeschrieben werden, damit die Daten weiterverarbeitet werden können.

```
xi = 1;
messdaten(:,xi) =
table2array(readtable('csvfile.csv', opts));
```

Jede Messung besteht aus einer Spalte Messwerten. Die Spalte wird von einer Tabelle in ein Array umgewandelt und der Variable *messdaten* zugeschrieben. Die Variable soll am Ende des Kalibrierprozesses insgesamt 17

Spalten haben. In jeder Spalte die Messwerte einer Gewichtsstufe. Hierfür wurde die Zahlvariable xi erstellt. Diese wird vor jedem Ausführen des Skriptes um eins inkrementiert.

Die Schritte zwei bis fünf müssen für jede Gewichtsstufe neu ausgeführt werden.

Für jede Messung wird erst das Gewicht auf die Waage gelegt und anschließend wird das MATLAB Skript gestartet. Pro Messung mit einer Messdauer von 20 Sekunden werden 42 Messdaten geliefert.

4.3 Datenweiterverarbeitung

Für die Datenverarbeitung müssen die Werte genauer betrachtet werden. Anhand der gemessenen Werte ist zu erkennen, dass die Waage ohne Gewicht einen Durchschnittswert von 479910 hat. Dieser Wert wird anschließend von allen anderen Werten abgezogen, damit 0 gleich 0 g sind.

Erstellt man pro Spalte der Gewichtsstufe einen Mittelwert, so erhält man 17 Werte. Damit nun eine Kennlinie erzeugt werden kann, wird eine Variable mit den Gewichtsstufen gebraucht.

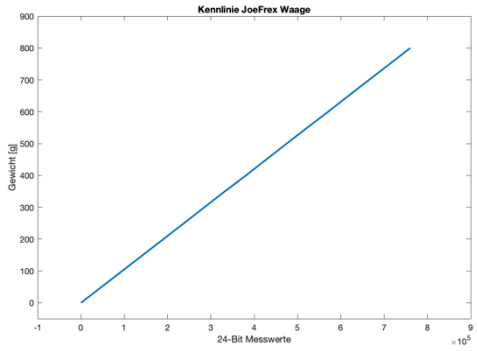

Abbildung 3: Kennlinie JoeFrex Waaage

Stellt man die beiden Vektoren gegenüber, erhält man eine Kennlinie der Waage, der sich von 0 Gramm bis 800 Gramm erstreckt.

Die Werte der Waage liegen auf der Abszisse, während die Gewichtswerte auf der Ordinate festgelegt sind. Zukünftig werden die Werte dieser Kennlinie dazu verwendet, die 24-Bit Werte der Waage in die physikalische Einheit Gramm umzuwandeln.

Für die Eignung der Waage im Messsystem wird abschließend eine statistische Auswertung durchgeführt.

5 Statistische Auswertung und Fazit

Im letzten Schritt wurde eine statistische Auswertung mit den Messwerten durchgeführt, um die Messgenauigkeit festzustellen.

Hierzu wurden die Abweichungen aller Gewichtsstufen betrachtet.

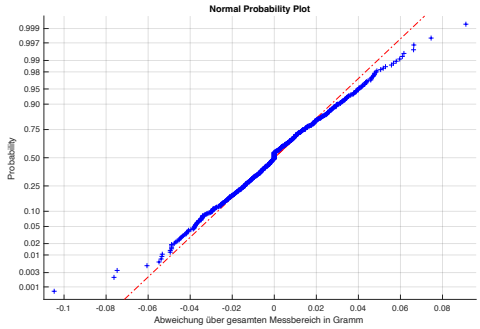

Abbildung 4: Häufigkeitsverteilung aller Messwertabweichungen

Daraus resultiert, dass keine hohen Messwertabweichungen vorhanden sind. Alle Messwerte befinden sich innerhalb von $\pm\, 0{,}1\ g$. Um die Aussage zu bestätigen, wurde zusätzlich eine Gaußsche Normalverteilung durchgeführt.

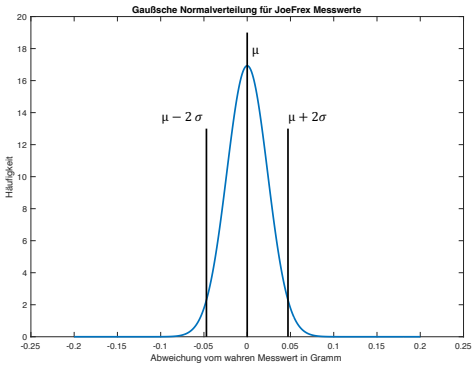

Abbildung 5: Normalverteilung der Messwerte

Mit der Normalverteilung wird der relevante Bereich bis zur doppelten Standardabweichung σ betrachtet. Dieser Bereich gibt eine Aussagewahrscheinlichkeit von 95,45 % wieder. Es ist zu erkennen, dass die Messwerte der Waage mit 95,45 % Wahrscheinlichkeit bis $\pm\, 0{,}05\ g$ liegen.

Damit ist die Waage für die Benutzung im Messsystem geeignet.

Abbildungsverzeichnis

Literaturverzeichnis

[1] A. Rohnen und T. Kuttner, „Deformatorische Aufnehmer," in *Praxis der Schwingungsanalyse*, Wiesbaden, Springer Vieweg, 2019, pp. 145-153.

[2] AVIA semiconductor, „reichelt.de," [Online]. Available: https://cdn-reichelt.de/documents/datenblatt/A300/HX711F_EN.pdf. [Zugriff am 29 12 2022].

[3] 29 12 2022. [Online]. Available: https://pinout.xyz.