



FH Aachen

Fachbereich Maschinenbau und Mechatronik

Modul

Datenmanagement und Leittechnik

Projektbericht

Prozessüberwachung einer Waschmaschine

vorgelegt von

Davin Tandrayuana
Matrikel-Nr.: 3551264

Referent:

Prof. Dr. Ing. Stephan Kallweit

Abgabedatum:

08.03.2023

Inhaltsverzeichnis

Abbildungsverzeichnis	3
1. Prozessüberwachung einer Waschmaschine.....	4
1.1. Aufgabenstellung.....	4
2.1. Ziel des Projekts.....	4
2. Erforderliche Ressourcen.....	5
3. Prozessbeschreibung.....	6
3. 1. Aufbau	6
3. 2. Programmablauf	6
3. 3. Ermittlung von der dominierenden Zeitkonstante.....	7
3. 4. Visualisierung der Daten in GUI (Qt)	7
4. Machine Learning Programm	8

Abbildungsverzeichnis

Abbildung 1: Verwendete Ressourcen.....	5
Abbildung 2: Verwendete Hardware zur Datenerfassung (Waschmaschine und IoT-Satz).....	6
Abbildung 3: Programmablauf	6
Abbildung 4: Qt-Benutzeroberfläche.....	7
Abbildung 5: Vergleich der Vorhersagengenauigkeit zwischen fünf verschiedenen Klassifizierungsverfahren	8
Abbildung 6: Confusion Matrix MLP Klassifikator	9

1. Prozessüberwachung einer Waschmaschine

Prozessüberwachung oder Zustandsüberwachung einer Maschine beschreibt eine regelmäßige oder permanente Erfassung des Maschinenzustandes durch Messung und Analyse physikalischer Größen, z. B. Schwingungen, Temperaturen, Lage/Näherung.

1.1. Aufgabenstellung

Im Rahmen des Projekts sollen verschiedene Betriebszustände einer Waschmaschine mit einem maschinellen Lernalgorithmus ermittelt und anschließend über eine grafische Benutzeroberfläche visualisiert werden. Mithilfe eines integrierten Beschleunigungssensors (MPU 6050) und Mikrocontrollers werden die Beschleunigungsdaten erfasst. Die Daten werden in einer cloudbasierten Datenbank (MongoDB) gesendet und gespeichert, sodass die Daten allen Teilnehmern zugänglich sind. Zunächst müssen die Daten mit maschinellen Lernalgorithmen trainiert werden. Anschließend werden die Daten mit der GUI (Qt) visualisiert.

2.1. Ziel des Projekts

Ziel ist es, dass die Studierenden sich mit den grundlegenden Aspekten der Prozessüberwachung und -diagnose vertraut werden und in der Lage sind, einen Prozess von der Datenerfassung bis zur Visualisierung der Untersuchungsergebnisse vollautomatisch zu analysieren und zu verstehen. Dafür sind verschiedene Kenntnisse erforderlich:

- Programmierung und Analyse einer Datenmenge (Python) und Anwendung von verschiedener Software (Jupyter Notebook, Thonny, MongoDB, etc.)
- Anwendung mit dem IoT-Gerät.
- Anwendung von einem Datenbanksystem zur Speicherung und Verwaltung der Daten.
- Theoretische Grundlage zur Bestimmung der Parameter, um saubere Datenerfassung zu ermöglichen.
- Generierung einer grafischen Benutzeroberfläche zur Darstellung der Prozessparameter und der Betriebszustände.

2. Erforderliche Ressourcen

Für dieses Projekt werden verschiedene Ressourcen (Hardware und Software) benötigt. Siehe Abbildung 1.


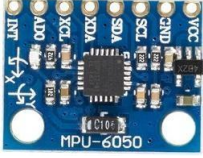




	Microcontroller ESP8266 zur Steuerung der Sensoren
	Beschleunigungssensor IMU MPU6050 für Erfassung von Beschleunigungsdaten
	PyCharm Entwicklungsumgebung für Python
	Thonny IDE für die Entwicklung der MicroPython
	Jupyter Notebook zur Vorbereitung der erfassten Daten und Trainieren des Klassifikators
	MongoDB und MongoDB Atlas zur lokalen und Cloud-Speicherung

Abbildung 1: Verwendete Ressourcen

3. Prozessbeschreibung

3. 1. Aufbau

Für die Erfassung der Beschleunigungsdaten wird das Breadboard mit dem Microcontroller und Beschleunigungssensor an der Waschmaschine befestigt mit Strom über einen Laptop versorgt (Abbildung 2).



Abbildung 2: Verwendete Hardware zur Datenerfassung (Waschmaschine und IoT-Satz)

3. 2. Programmablauf

Zunächst wird der programmierte Code über Thonny IDE in den Mikrocontroller geflasht. Es gibt 4 Codes, die im Mikrocontroller gespeichert werden müssen (main.py, imu.py, vector3d.py und wifi.py). Sobald alle Codes erfolgreich geflasht sind, beginnt die Datenerfassung von der Waschmaschine. Das Breadboard mit dem IoT-Gerät wurde auf die Oberseite der Waschmaschine geklebt, damit keine Dämpfung erzeugt wird.

Die von der MPU6050 erfassten Daten werden kodiert und über das TCP-Protokoll an den Laptop gesendet. Dort werden sie entschlüsselt und in die Werte "Ax", "Ay" und "Az" zerlegt. Die Daten werden dann an die Collection in MongoDB (Cloud-Anbieter) gesendet und dort gespeichert.

Die in der Cloud gespeicherten Daten werden importiert und in einem Jupyter-Notebook analysiert. Zunächst müssen die Daten aufbereitet werden, damit nur saubere Daten für das Training des maschinellen Lernalgorithmus verwendet werden. Sobald das Modell fertig trainiert ist, kann es verwendet werden, um den Zustand der Waschmaschine zu bestimmen. Anschließend werden die Beschleunigungsdaten in einer grafischen Benutzeroberfläche (Qt) visualisiert.



Abbildung 3: Programmablauf

3. 3. Ermittlung von der dominierenden Zeitkonstante

Um den Aliasing-Effekt zu vermeiden, sollte die höchste zu erwartende Signalfrequenz bestimmt und die Abtastfrequenz darauf abgestimmt werden. Es soll mindestens Doppelte der höchsten Frequenz abgetastet werden, damit das Eingangssignal rekonstruiert werden kann. Die maximale Drehzahl der Waschmaschine beträgt 1400 min⁻¹.

$$f_{max} = \frac{n_{max} \left[\frac{1}{min} \right]}{60 \left[\frac{s}{min} \right]} = \frac{1400 \left[\frac{1}{min} \right]}{60 \left[\frac{s}{min} \right]} = 23,33 [Hz]$$

Die maximale Drehfrequenz des Waschprogramms beträgt dementsprechend 23,33 Hz. Es ist aber dennoch sinnvoll, die Abtastfrequenz höher als die doppelte Frequenz des Eingangssignals anzusetzen. Um eine möglichst genaue Abtastung des Eingangssignals zu erreichen, wurde entschieden, mit einer Frequenz von 100 Hz abzutasten.

3. 4. Visualisierung der Daten in GUI (Qt)

In diesem Projekt wird eine Qt-Benutzeroberfläche zur Visualisierung der Betriebszustände und Beschleunigungswerte verwendet. Das Hauptfenster wurde mit dem PySide6-Modul entworfen, das die Verbindung mit dem Mikrocontroller ermöglicht, um den Programmablauf, die Betriebszustände und die erfassten Daten zu visualisieren. Die entworfene Benutzeroberfläche ist in Abbildung 4 dargestellt.

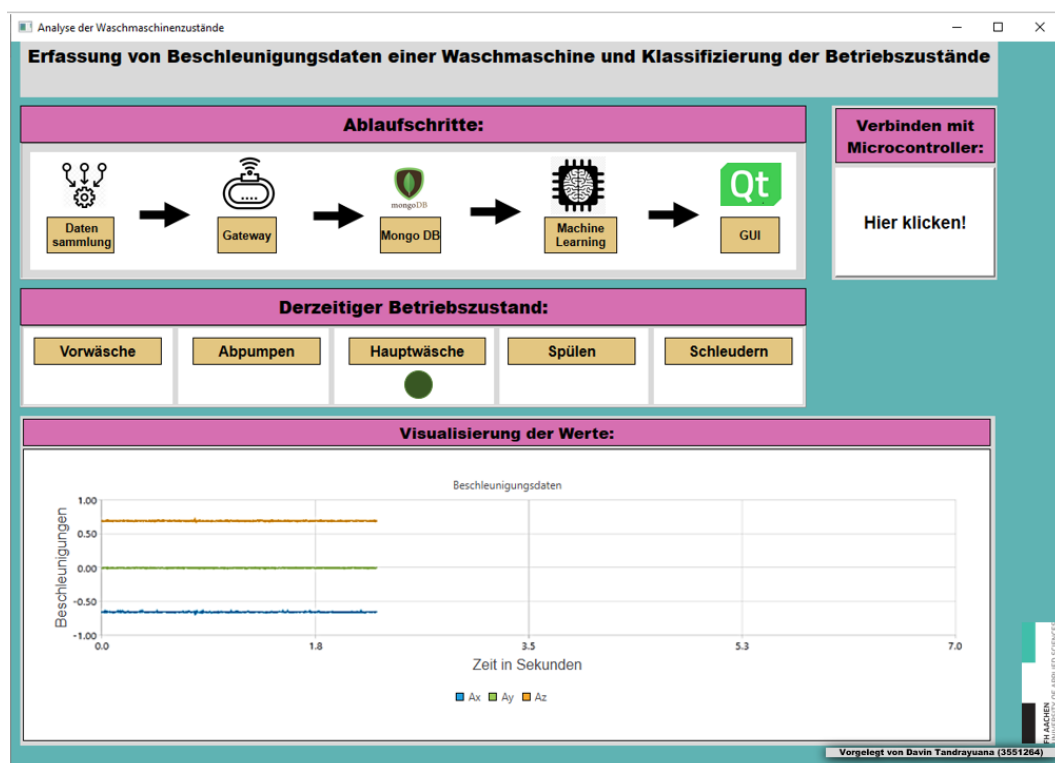


Abbildung 4: Qt-Benutzeroberfläche

4. Machine Learning Programm

In diesem Projekt wird zunächst eine Trainingsdatei (`training_data.csv`) erstellt. Das Modell für maschinelles Lernen muss zuerst trainiert werden und die gesammelten Beschleunigungsdaten werden als Eingabe verwendet. Die Eingabedaten wurden von MongoDB Cloud heruntergeladen und dann in Jupyter Notebook importiert. Das Jupyter Notebook ist in 5 Kapitel unterteilt. In den Kapiteln 1 und 2 werden die Rohdaten analysiert und visualisiert. Kapitel 3 beschreibt die Aufbereitung der Daten für die Klassifikatoren. In Kapitel 4 werden die Vorbereitungsdaten trainiert. In Kapitel 5 wird dann das trainierte Modell mit den Rohdaten getestet.

Um die Betriebszustände so genau wie möglich vorherzusagen, werden fünf verschiedene Klassifizierungsmodelle getestet und verglichen, nämlich:

- MLP – Multi Layer Perceptron
- Decision Tree
- Linear Regression
- SVM – Support Vector Machine
- Perceptron

Bei der Prozessaufnahme soll der Betriebszustand anhand von 100 absoluten Geschwindigkeitswerten und deren Frequenzspektrum identifiziert werden. Daher wird der Klassifikator mit einem Amplitudendatensatz, der 100 Spalte der Amplituden und 1 Spalte des Status beinhaltet, trainiert. Um verschiedene Klassifikatoren richtig vergleichen zu können, muss für jeden Klassifikator der gleiche Datensatz verwendet werden. Da jede Kategorie eine unterschiedliche Anzahl von Datensätzen enthalten kann, ist die maximal zu verwendende Datensatzgröße auf die Größe des Datensatzes mit der geringsten Anzahl von Einträgen begrenzt. Nach dem Vergleich der fünf Klassifikatoren zeigt sich, dass MLP die höchste Genauigkeit und Stabilität der Ergebnisse aufweist (Abbildung 5). Außerdem zeigt der MLP-Klassifikator ein gutes Ergebnis in der Konfusionsmatrix (Abbildung 6).

```
Status Description:  
0 = Waschen  
1 = Pumpen  
2 = Wasserzu- und ablauf  
3 = Schleudern  
4 = Ausgeschaltet
```

```
Datasheet_status:  
[3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,  
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,  
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,  
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,  
3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]
```

```
Status prediction from machine learning:  
MLP: ['3']    Desc.Tree: ['1']    LR: ['0']    SVM: ['0']    PPN: ['1']
```

Abbildung 5: Vergleich der Vorhersagengenauigkeit zwischen fünf verschiedenen Klassifizierungsverfahren

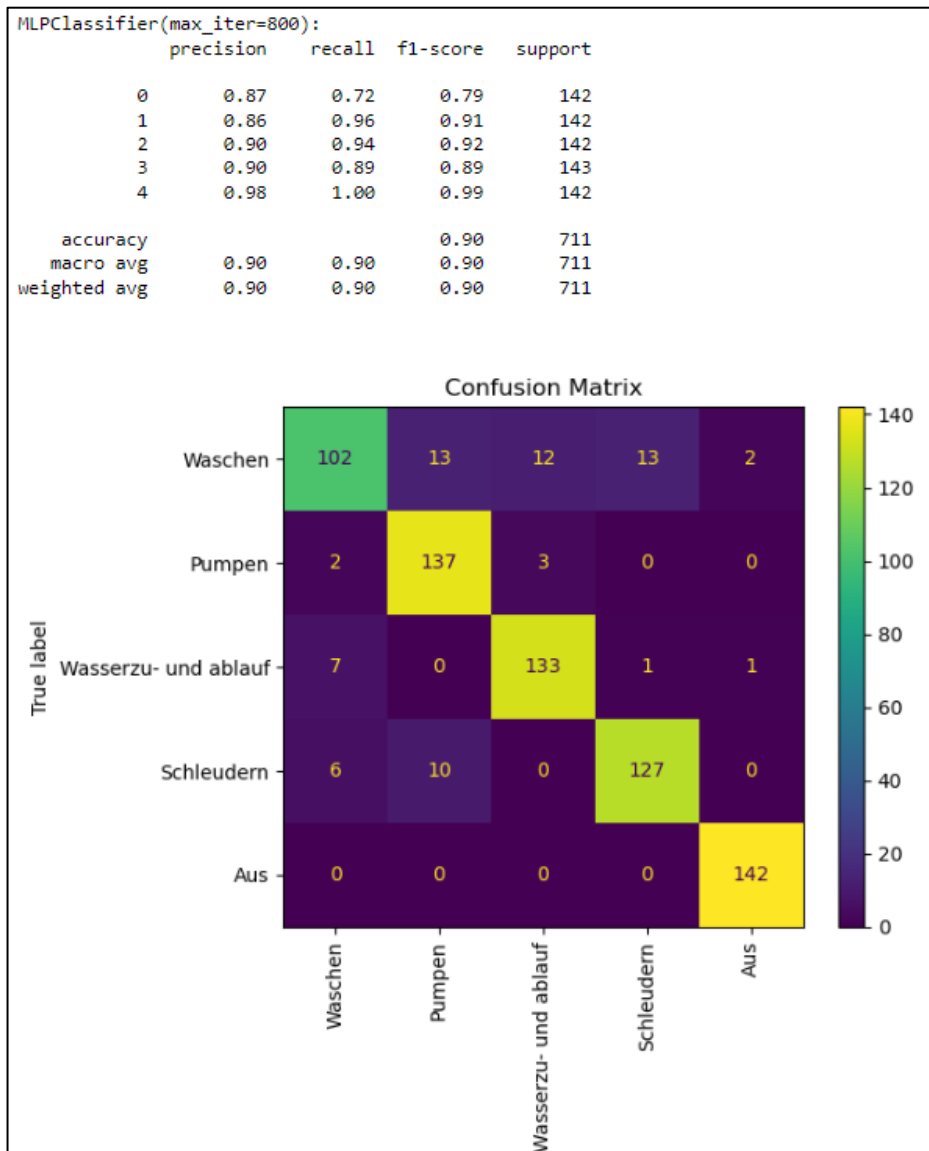


Abbildung 6: Confusion Matrix MLP Klassifikator