Bachelorarbeit - Thesis

Kurzzeitprognose der Turmschwingungskinematik von Onshore-Windenergieanlagen

Autor:	Zelgai Nemati
Matrikelnummer:	4516359
Studiengang:	B. Sc. Systems Engineering
Erstprüfer:	Prof. Dr. Klaus-Dieter Thoben
Zweitprüfer:	M. Sc. Andreas Haselsteiner
Betreuer Contact Software:	Dr. Nicole Göckel
	Dr. Thomas Dickopf
Abgabedatum:	24.03.2022

INHALTSVERZEICHNIS

Formelzeichen

Zeichen	Erklärung
y_t	Prognosewert zum Zeitpunkt t
μ	Arithmetisches Mittel
ϕ	Autoregression Gewichtungsparameter
i	Anzahl der Lags
e_t	Fehler des Prognosewerts zum Zeitpunkt t
θ	Moving-Average Gewichtungsparameter
p	Anzahl Autoregressions-Terme
d	Anzahl Integrations-Terme
q	Anzahl Moving-Average-Terme
P	Anzahl saisonaler Autoregressions-Terme
D	Anzahl saisonaler Integrations-Terme
Q	Anzahl Moving-Average-Terme
$g(t)$	Prophet Trendfunktion
$s_p(t)$	Prophet Saisonalitätsfunktion
$h(t)$	Prophet Interpolationsfunktion
g	Erdbeschleunigungskonstante
$a(t)$	Beschleunigung
$v(t)$	Geschwindigkeit
$s(t)$	Strecke
$\hat{y}_i$	Prognosewert für den Zeitpunkt t
RMSE	Root Mean Square Error
σ	Standardabweichung

1 Einleitung

Onshore-Windenergie liefert schon heute einen substanziellen Teil des Energiemixes und hat in den letzten zehn Jahren erhebliche Fortschritte gemacht. Beispiele sind größere und zuverlässigere Turbinen, steigende Bauhöhen und größere Rotorblattdurchmesser [1][2][3]. Aufgrund dieses technologischen Fortschritts und der optimierten Skalierung konnten zwischen den Jahren 2010 und 2019 die Stromgestehungskosten um 39% (von 0.086 USD/kWh auf 0.053 USD/kWh) und die Installationskosten um 24% gesenkt werden [4]. Diese Entwicklung führt zu einer gesteigerten Wettbewerbsfähigkeit der Windenergie Branche und somit haben 75% aller im Jahr 2019 in Auftrag gegebenen Windprojekte niedrigere Stromgestehungskosten als die billigste fossile Energiequelle [4]. Die Windenergie ist auf dem Weg eine tragende Säule des zukünftigen grünen Energiemixes zu werden [1][2][3][4].

Heutzutage werden die meisten Offshore Windenergieanlagen komponentenweise installiert und um die Kosten weiterhin zu senken, ist es insbesondere notwendig, den Installationsprozess zu optimieren [1][2][3]. Die Montage der Rotorblätter stellt dabei die größte Herausforderung dar, denn hier ist hohe Präzision und Sorgfalt erforderlich, um das Blattende in die Rotornabe einzusetzen [1][2][5][6]. Der Wind übt Lasten auf die mechanischen Strukturen der Windkraftanlage aus und die daraus resultierenden Relativbewegungen zwischen Turm und den Rotorblättern erschweren die Blattmontage [1][2][5][6]. Überschreitet die Relativbewegung einen definierten Schwellenwert, kann die Installation nicht mehr durchgeführt werden, da Schäden beim Montagevorgang zu erwarten sind und es kommt zu einer kostspieligen Verzögerung [1][2][5][6].

Deshalb werden aktuell Wetterlimits zur Planung solcher Installationsmaßnahmen verwendet, wobei eine direktere limitierende Größe als das Wetter, eine Prognose der Turmschwingung wäre. Jedoch ist bislang, aus wissenschaftlicher Perspektive, nicht beantwortet, welche Modelltypen am besten geeignet sind, mit welcher Genauigkeit sich eine solche Prognose der Turmschwingungskinematik verwirklichen lässt und wie diese Erkenntnisse optimal in der Praxis genutzt werden können z.B. in Form einer Echtzeit IoT Anwendung.

2 Zielstellung

Aktuell werden wie beschrieben Wetterlimits bei der Planung und Durchführung von Installationsvorgängen verwendet, da bei einer zu hohen Relativbewegung zwischen Turm und Rotorblatt eine Installation nicht möglich ist und es zu Schäden an den Komponenten kommen kann [1][2][5][6]. Eine direktere limitierende Größe als das Wetter, wäre eine Prognose der Turmschwingung (in Abbildung 2-1 blau markiert). Deswegen soll im Rahmen dieser Abschlussarbeit die Forschungsfrage beantwortet werden, wie genau die Schwingungskinematik eines Windenergieanlagen-Turms für die nächsten Sekunden und Minuten vorhergesagt werden kann.

Es sollen drei verschiedene Prognosemodelle (ARIMA, SARIMA, Prophet) implementiert werden, welche auf GitHub unter einer MIT Lizenz frei verfügbar seien sollen. Die Genauigkeit dieser Modelle wird dann für verschiedene Zeiträume getestet, sodass eine fundierte Beantwortung der wissenschaftlichen Fragestellung sich aus diesen Ergebnissen ableitet. Diese optimierten Prognosemodelle, inklusive einer Datenpipeline, welche einen kontinuierlichen Messdatenstrom verarbeitet, sollen darauffolgend in der Contact Elements for IoT Plattform entwickelt werden. Durch Oberflächenkonfiguration soll es ebenfalls möglich sein diese Modelle auf andere Datensätze und Fragestellungen zu adaptieren.

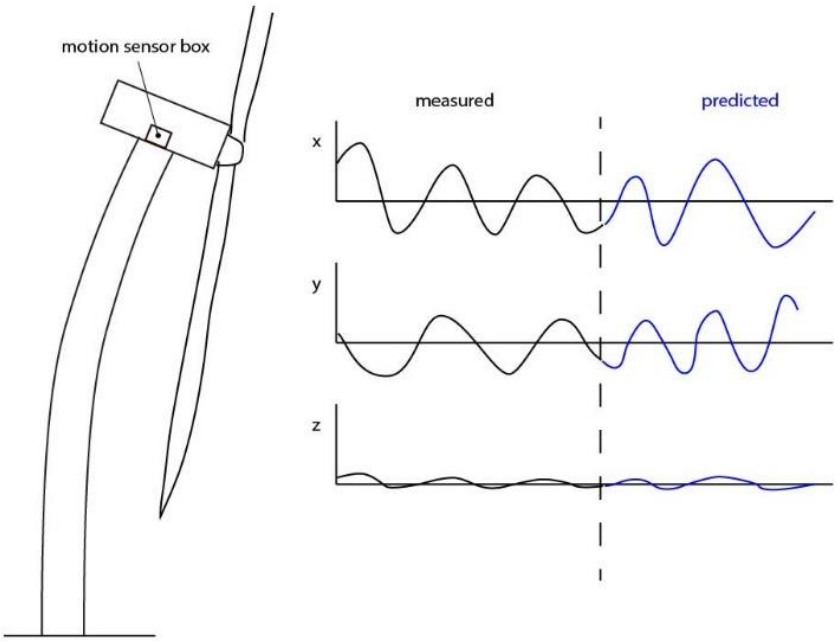

Abbildung 2-1 Prognose der Turmschwingungskinematik

3 Stand der Forschung

3.1 Turmschwingungskinematik

Die Analyse der Turmschwingungskinematik ist notwendig, um den Installationsprozess von Windenergieanlagen zu optimieren. Im Jahr 2020 wurden im Rahmen einer Messkampagne Daten zur Turmkinematik, während der Installation von Offshore Windenergieanlagen, erhoben [5]. Die Auswertung dieser Positionsdaten hat ergeben, dass die Schwingungskinematik des Turms eine besondere orbitförmige Charakteristik aufweist, welches ebenfalls der nachfolgenden Abbildung 3-1 zu entnehmen ist [1][2][5].

Die Turmschwingungskinematik wird maßgeblich vom Wind und anderen Umweltparametern beeinflusst, die in der einschlägigen Literatur häufig als „zufällige“ Störgrößen bzw. stochastische Prozesse aufgefasst werden. Im Rahmen dieser Abschlussarbeit wird von einem stationären Datensatz ausgegangen, da der Erwartungswert und die Varianz nicht zeitabhängig sind und keine Saisonalität vorliegt [7].

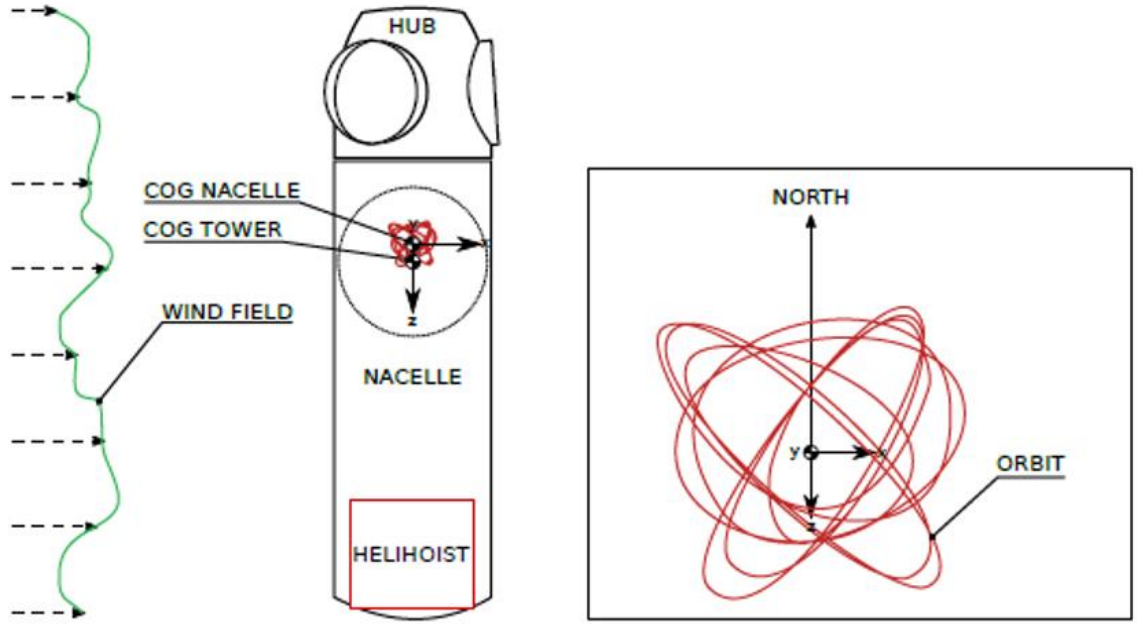

Abbildung 3-1 Auswertung der Installationsdaten vom Offshore Windpark: ianel Windpark Borkum 2“ [5]

3.2 Prognosemodelle

Um eine fundierte Auswahl der zu implementierenden Prognosemodelle zu gewährleisten, werden im Folgenden konkrete Kriterien aus der Charakteristik der Turmschwingungskinematik und der vorliegenden Problemstellung hergeleitet.

Kriterium 1

Wie bereits im Kapitel 3.1 beschrieben wurde, wird davon ausgegangen, dass es sich bei den Umwelteinflüssen um zufällige Störgrößen bzw. stochastische Prozesse handelt. Deshalb sollte ein gewähltes Prognosemodell dazu in der Lage sein sich zeitlich verändernde Eigenschaften des Schwingungssignals berücksichtigen zu können.

Kriterium 2

Da es sich bei den Umwelteinflüssen um zufällige Störgrößen bzw. stochastische Prozesse handelt, kann keine Aussage über das auftretende Rauschen der Zeitreihendaten getroffen werden. Deshalb sollte das gewählte Prognosemodell keine besonderen Anforderungen an das Rauschverhältnis stellen.

Kriterium 3

Nachdem das Modell validiert und die notwendigen Prognoseparameter optimiert wurden, sollte die Berechnung sehr schnell sein, speziell vor dem Hintergrund der Contact Elements for IoT Integration.

Kriterium 4

Das gewählte Prognosemodell sollte erfahrungsgemäß präzise Ergebnisse mit einer hohen Genauigkeit, für die Strukturkinematikprognose liefern.

3.2.1 Autoregression (AR)

Autoregressive Prognosemodelle, abgekürzt AR-Prognosemodelle, stellen die einfachste Schätzmethode dar und können nur auf stationäre Datensätze angewandt werden [7][8]. AR-Modelle prognostizieren zukünftige Parameterwerte, anhand der vergangenen Werte, welche im Fachjargon bzw. in der einschlägigen Literatur auch als „Lags“ bezeichnet werden [7][8]. Diese Abhängigkeit zu vergangenen Werten (bzw. der Einfluss der vergangenen auf zukünftige Werte) kann der nachfolgenden Abbildung 3-2 entnommen werden.

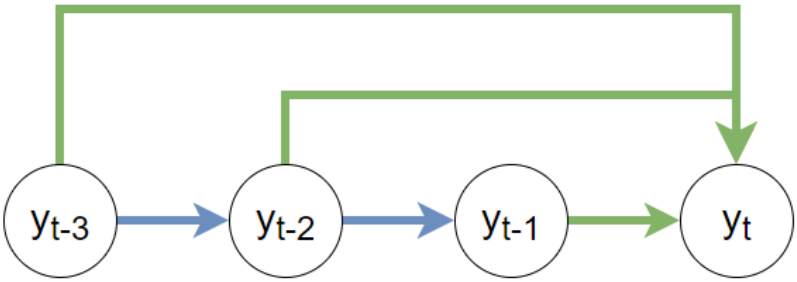

Abbildung 3-2 AR-Modell Funktionsweise

Ein AR Modell erster Ordnung bzw. *AR(1)* Modell wird durch die folgende Formel beschrieben [7][8]:

$$y_t = \mu + \phi * y_{t-1} + e_t \tag{3-1}$$

Für dieses Autoregressionsmodell mit dem Grad 1, entspricht der Parameterwert zum Zeitpunkt *t*, der Summe aus dem Mittelwert, dem gewichteten Parameterwert zum Zeitpunkt *t-1* und einem Fehlerwert *e*, welcher in der einschlägigen Literatur auch als „White-Noise“ bezeichnet wird [7] [8].

Diese mathematische Definition des eingeführten Prognosemodelltyps, lässt sich auch auf beliebig viele Lags erweitern und ist somit nicht wie in Formel 3-1 auf einen Lag beschränkt. Eine allgemeine Definition für *i* Lags, ist der Formel 3-2 zu entnehmen [7][8]:

$$y_t = \mu + \left(\sum_{j=0}^{i} \phi * y_{t-i} \right) + e_t \tag{3-2}$$

Bei der Auswahl des AR Modells bzw. bei der Bestimmung des optimalen Komplexitätsgrads gibt es einen Grundsatz. Dieser Grundsatz besagt, dass wenn zwei Prognosemodelle die beinahe gleiche Genauigkeit der Prognose hervorbringen, das Modell mit dem niedrigeren Grad bzw. der niedrigeren Komplexitätsstufe verwendet wird [7][8].

Diese Abwägung zwischen einer möglichst niedrigen Komplexität und einer möglichst hohen Genauigkeit des Prognosemodells und der sich daraus ableitenden Fragestellung, welche Lags mit in die Prognose einbezogen werden sollen, wird in späteren Kapiteln dieser Abschlussarbeit automatisch durch die verwendeten Algorithmen kalibriert.

Im Rahmen dieser Abschlussarbeit wird kein AR Prognosemodell verwendet, da das ARIMA Modell (welches im Kapitel 3.2.3 theoretisch eingeführt wird) auf diesem aufbaut und erhebliche Vorteile gegenüber diesem Modelltyp hat.

3.2.2 Moving-Average (MA)

Moving-Average Prognosemodelle, abgekürzt MA-Prognosemodelle, sind ebenso wie AR-Modelle nicht sehr komplex und können ebenfalls nur auf stationäre Datensätze angewandt werden [7][8]. MA-Modelle prognostizieren nicht anhand der vergangenen Parameterwerte (so wie es AR-Modelle machen), sie prognostizieren anhand der vergangenen Fehlerwerte [7][8]. Dazu kann man sich zwei Graphen vorstellen. Einer bildet den tatsächlichen Verlauf ab und der andere bildet die Prognose dieser Werte ab. Nun wird für die Lags die Differenz aus Prognosewert und tatsächlichem Wert berechnet. Kalkuliert wird ein Fehlerwert für jeden zu betrachtenden Lag. Für den initialen Wert einer Zeitreihe wird als Prognosewert der Durchschnitt aller Parameterwerte genommen. Dieses Vorgehen ist der folgenden Abbildung 3-3 zu entnehmen, wo die tatsächlichen Werte in blau, die Prognosewerte in Rot und die errechneten Fehlerwerte in orange dargestellt sind.

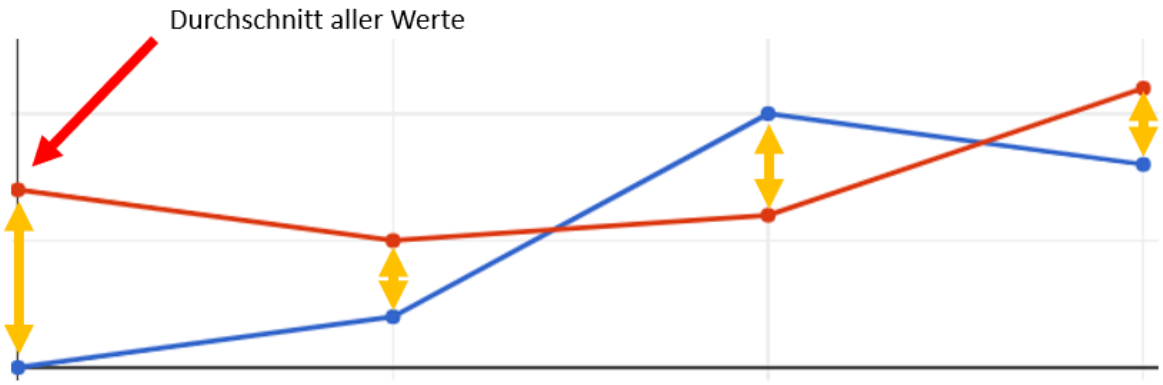

Abbildung 3-3 MA Bestimmung des initialen Prognosewerts und der folgenden Fehlerwerte

Wie bereits bei dem AR Modell, kann ein MA Modell beliebig viele Lags zur Prognose verwenden und es kann somit ein beliebig hoher grad gewählt werden. Ein *MA(i)* Modell, wird durch die folgende Formel 3-3 beschrieben [7][8]:

$$y_t = \mu + \left(\sum_{j=0}^{i} \theta * e_{t-i} \right) + e_t \tag{3-3}$$

Im Rahmen dieser Abschlussarbeit wird ebenfalls kein MA Prognosemodell verwendet, da das ARIMA Modell (welches im Kapitel 3.2.3 theoretisch eingeführt wird) auf diesem aufbaut und erhebliche Vorteile gegenüber diesem Modelltyp hat.

3.2.3 ARIMA

Autoregressive Integrated Moving Average Prognosemodelle, abgekürzt ARIMA-Prognosemodelle, sind eine Kombination der beiden schon thematisierten Modelltypen [9][10][11][12][13]. AR-Modelle und MA-Modelle stehen in einem besonderen Zusammenhang, der im Folgenden aus der Formel 3-1, welche ein *AR(1)* Modell beschreibt, mathematisch hergeleitet wird:

$$y_t = \mu + \phi * \boldsymbol{y_{t-1}} + e_t \tag{3-1}$$

Ausschreiben des blau markierten Parameters $\boldsymbol{y_{t-1}}$

$$y_t = \mu + \phi^2 * \boldsymbol{y_{t-2}} + \phi * e_{t-1} + e_t$$

Ausschreiben des blau markierten Parameters $\boldsymbol{y_{t-2}}$

$$y_t = \mu + \phi^3 * y_{t-3} + \phi^2 * e_{t-2} + \phi * e_{t-1} + e_t$$

...

$$y_t = \mu + \phi^t * y_1 + \phi^{t-1} * e_2 + \phi^{t-2} * e_3 + \cdots + e_t \tag{3-4}$$

Die hergeleitete Formel 3-4 entspricht einem *MA (∞)* Modell, also einer Summe gewichteter Fehlerwerte, was der Formel 3-3 entnommen werden kann. Somit besteht der Zusammenhang *AR (1) = MA (∞)*.
Diese beschriebene Beziehung der beiden Modelltypen (AR und MA) ermöglicht es einem Prognosemodelle mit erhöhter Genauigkeit und verringerter Komplexität zu implementieren, indem man die einzelnen Terme eines AR und MA Modells in einem gemeinsamen ARIMA Modell kombiniert [9][10][11]. Diese Kombination der AR und MA Terme kann der folgenden Formel 3-5 beispielhaft entnommen werden [9][10][11][12][13]:

$$y_t = \mu + \phi * y_{t-1} + \cdots + \phi * y_{t-p} + \theta * e_{t-1} + \cdots + \theta * e_{t-q} + e_t \tag{3-5}$$

Anders als bei den bisher behandelten Prognosemodellen, gibt es beim ARIMA Modell drei Grad-Parameter, welche in der einschlägigen Literatur *p*, *d* und *q* genannt werden [9][10][11][12][13]. Die Notation für diesen Modelltyp kann der folgenden Formel 3-6 entnommen werden:

$$ARIMA(p, d, q) \tag{3-6}$$

Der Parameter *p* beschreibt den Grad des AR-Teils, der Parameter *d* beschreibt den Grad des I-Teils und der Parameter *q* beschreibt den Grad des MA-Teils [9][10][11][12][13].

Somit wird ein *ARIMA(1, 0, 1)* Modell, mit einem AR-Teil, keinem I-Teil und einem MA-Teil, durch die folgende Funktion 3-7 beschrieben:

$$y_t = \mu + \phi * y_{t-1} + \theta * e_{t-1} + e_t \tag{3-7}$$

Die thematisierte Komplexitätsreduktion führt dazu, dass nach der Validierung der Prognoseparameter eine schnelle Kalkulation der Prognose möglich ist [10][11]. Ein weiterer entscheidender Vorteil von ARIMA und SARIMA (siehe Kapitel 3.2.4) Modellen ist, dass sie sowohl auf stationäre als auch auf nicht stationäre Datensätze anwendbar sind, welches durch eine optionale Integration der Zeitreihe ermöglicht wird (dafür steht das *I* im Namen des Modeltyps) [10][11]. Außerdem berechnen beide Modelle die Prognose auf Grundlage der Vergangenheitswerte (und der vergangenen Fehlerwerte) und gewichten die zeitlich näherliegenden Werte stärker als Werte, die lange in der Vergangenheit zurück liegen. Dadurch ist sichergestellt, dass die Modelle auf sich ändernde Eigenschaften des Schwingungssignals optimal reagieren können [10].

Im Kapitel 7.1.1, indem die praktische Umsetzung des ARIMA Modells thematisiert wird, wird ein bereits bestehender Algorithmus verwendet, um das optimale ARIMA Modell, passend zur jeweiligen Fragestellung und der zu analysierenden Zeitreihe, zu identifizieren.

3.2.4 SARIMA

Seasonal Autoregressive Integrated Moving Average Prognosemodelle, abgekürzt SARIMA-Prognosemodelle, sind eine Weiterentwicklung von ARIMA-Modellen [12][14][15][16][17][18]. SARIMA Prognosemodelle beziehen zusätzliche saisonale Parameter zur Prognose mit ein und können somit für bestimmte Zeitreihen optimierte Ergebnisse hervorbringen, weshalb sie sich einer hohen Beliebtheit erfreuen [14][16]. Besonders vor dem Hintergrund, der im Rahmen dieser Abschlussarbeit zu behandelnden Fragestellung, bei der die Turmkinematik durch Umweltparameter maßgeblich beeinflusst wird, sind durch das SARIMA Modell Prognoseergebnisse mit einer hohen Genauigkeit zu erwarten, da diese einer Saisonalität unterliegen.
Die Notation für diesen Modelltyp kann der folgenden Formel 3-8 entnommen werden [14][15][16]:

$$SARIMA(p,d,q)(P,D,Q)_s \tag{3-8}$$

Das *P* steht für die Anzahl der saisonalen AR Terme, *D* für die Anzahl der saisonalen I Terme, *Q* für die Anzahl der Saisonalen MA Terme und *s* für die Länge der Saisonalität [14][15][16]. Der folgenden Formel 3-9 kann eine allgemeine Definition des SARIMA Modells entnommen werden [14][15][16]:

$$y_t = \mu + \left(\sum_{j=0}^{p} \phi * y_{t-i} + \sum_{j=0}^{P} \phi * y_{t-s*j}\right) + \left(\sum_{j=0}^{d} \theta * e_{t-i} + \sum_{j=0}^{D} \theta * e_{t-s*j}\right) + e_t \tag{3-9}$$

Die Vorteile des ARIMA Prognosemodells, welche im Kapitel 3.2.3 thematisiert wurden, gelten ebenso im Kontext des SARIMA Modells. Außerdem sind wie beschrieben durch die saisonale Komponente bessere Ergebnisse zu erwarten. Diese Vermutung wird dann im Kapitel 7.4 abschließend beantwortet.
Im Kapitel 7.2.1, indem die praktische Umsetzung des SARIMA Modells thematisiert wird, wird ein bereits bestehender Algorithmus verwendet, um das optimale SARIMA Modell, passend zur jeweiligen Fragestellung und der zu analysierenden Zeitreihe, zu identifizieren.

3.2.5 Prophet (Neuronales Netzwerk)

Um die Eingangs formulierte Fragestellung, wie genau die Turmkinematik für die nächsten Sekunden und Minuten vorhergesagt werden kann, fundiert beantworten zu können, sollen im Rahmen dieser Arbeit nicht nur statistische Methoden, sondern auch Neuronale Netze zur Kurzzeitprognose der Turmschwingungskinematik verwendet werden, da diese eventuell bessere Ergebnisse hervorbringen.
Prophet ist ein vom Unternehmen Facebook entwickeltes Zeitreihen-Prognose-Tool bzw. Framework, welches open-source ist und mithilfe der Programmiersprachen R und Python genutzt werden kann [19][21][22]. Eine detaillierte Dokumentation wurde vom Großkonzern erstellt und ist unter folgendem Link aufrufbar:

https://facebook.github.io/prophet/docs/quick_start.html#python-api

Die Analyse bzw. Prognose durch Prophet basiert auf einem neuronalen Netz, welches aus drei Grundfunktionen zusammengesetzt ist [19][20][22]. Dies kann der folgenden Formel 3-10 entnommen werden:

$$Y(t) = g(t) + s_p(t) + h(t) + e_t \tag{3-10}$$

Der grundlegende Trend einer Zeitreihe wird durch g(t) modelliert, die mögliche Saisonalität einer Zeitreihe wird durch s(t) dargestellt und h(t) steht für mögliche Lücken im Datensatz bzw. in der zu analysierenden Zeitreihe [19][20][22]. Dadurch ist es möglich das initial für wirtschaftswissenschaftliche Fragestellungen entwickelte Netz (ähnlich wie beim ARIMA Modell) auf diverse Fragestellungen zu adaptieren [21].
Im Rahmen dieser Arbeit wird Prophet gegenüber anderen Neuronalen Netzwerken bevorzugt, da es hoch konfigurierbar ist und keine besonderen Anforderungen an den Datensatz in Bezug auf Stationarität stellt [19][20][21][22]. Ein weiteres ausschlaggebendes Argument ist, dass es mithilfe von Prophet möglich ist die Prognose in Echtzeit zu testen und optimieren, sodass es besonders in IoT Szenarien zu besseren Ergebnissen führt [21].

4 Senvion Windkraftanlage

Die Senvion Windkraftanlage, mit einer Nennleistung von 3.37MW, dient als Forschungsplattform zur Entwicklung und Erprobung praxistauglicher Lösungen in der Anlagentechnik, der Erprobung neuartiger Sensoren zur Überwachung von Getriebe, Triebstrang und Rotorblatt und liefert Messdaten für Verbesserungen in Konstruktion, Werkstoffwahl, Fertigung und Steuerung von Windenergieanlagen.
Der nachfolgenden Abbildung 4-1, der Windkraftanlage, sind Größenparameter und die Position der Sensorboxen zu entnehmen. Der konkrete Aufbau, Funktionsweise und Daten, welche von der Sensorbox, erfasst werden, werden in dem Kapitel 5 thematisiert.

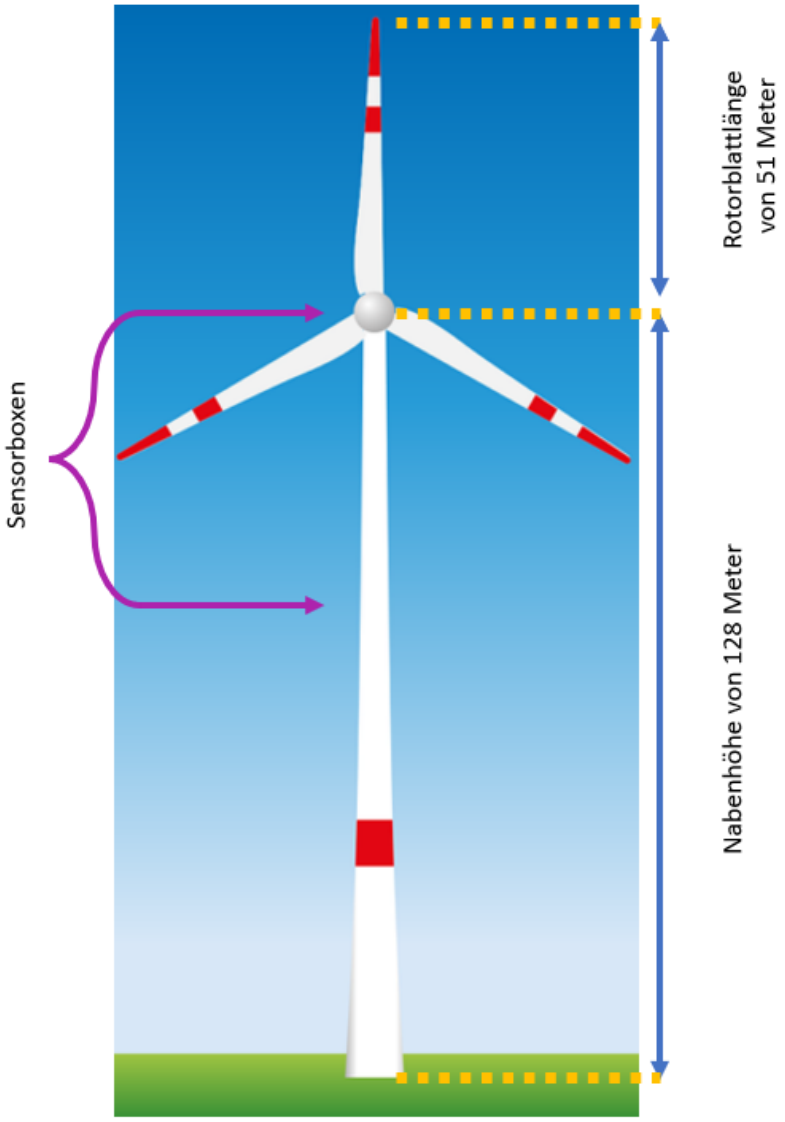

Abbildung 4-1 Skizze der Senvion Windkraftanlage

5 Flucto Sensorbox

Im Kapitel 4 wurde bereits beschrieben, dass an der Senvion Windkraftanlage zwei Sensorboxen der Firma Flucto GmbH angebracht sind, welche verschiedenste Parameter aufnehmen. Der folgenden Abbildung 5-1 ist eine solche Sensorbox zu entnehmen.

Abbildung 5-1 Flucto Sensorbox

Diese Boxen basieren auf einem Raspberry Pi Zero W und sind aufgrund der hohen Anzahl an Schnittstellen und verbauten Sensoren in einer Vielzahl von Messprojekten einsetzbar. Unter dem folgenden Link, der zu einem GitHub Repository führt, kann detaillierte Information zur Hardware und Software der beschriebenen Sensorbox abgerufen werden:

https://github.com/flucto-gmbh/motion-sensor-box

Essenziell für die Strukturierung der Prognosemodelle, der Entwicklung der Datenpipeline und der Contact Elements for IoT Integration ist das grundlegende Format der Messdaten, welche im Folgenden als „Rohdaten“ bezeichnet werden. Diese liegen im JSON-Format vor. Dabei beschreibt jedes Objekt die Messdaten, die zu einem gegebenen Zeitpunkt aufgenommen wurden und es wird grundlegend zwischen IMU Objekten, die die zu analysierenden Messdaten beinhalten und GPS Objekten, welche zur Positionskalibrierung dienen unterschieden. Der nachfolgenden Abbildung 5-2 kann dieses beschriebene Schema entnommen werden.

```
{
        "imu" : [timestamp, uptime, acc_x, acc_y, acc_z, rot_x, rot_y, rot_z,
        mag_x, mag_y, mag_z, temp]
}
{
        "gps" :
}
```

Abbildung 5-2 Rohdatenstruktur

Dem folgenden Link können zwei Rohdatensätze entnommen werden, die grundlegend im Rahmen dieser Abschlussarbeit als Testdatensätze verwendet wurden:

https://github.com/WindIO-Bachelorthesis/Shortterm_Forecast/tree/main/data/raw

Im folgenden Kapitel 6, werden dann diese thematisierten Rohdaten, welche von den Sensorboxen aufgenommen werden mithilfe einer Datenpipeline verarbeitet, sodass diese dann als Eingabe der Prognosemodelle und für die CE4IoT Integration verwenden können.

6 Datenpipeline

Die im Kapitel 5 beschriebenen Rohdaten müssen verarbeitet werden, um als Eingabe der Prognosemodelle und der CE4IoT Integration genutzt werden zu können. Die Datenpipeline orientiert sich an der Datenverarbeitung, welche in diesen Studien [1][2][5] thematisiert wird. Der folgenden Abbildung sind die einzelnen Verarbeitungsschritte zu entnehmen:

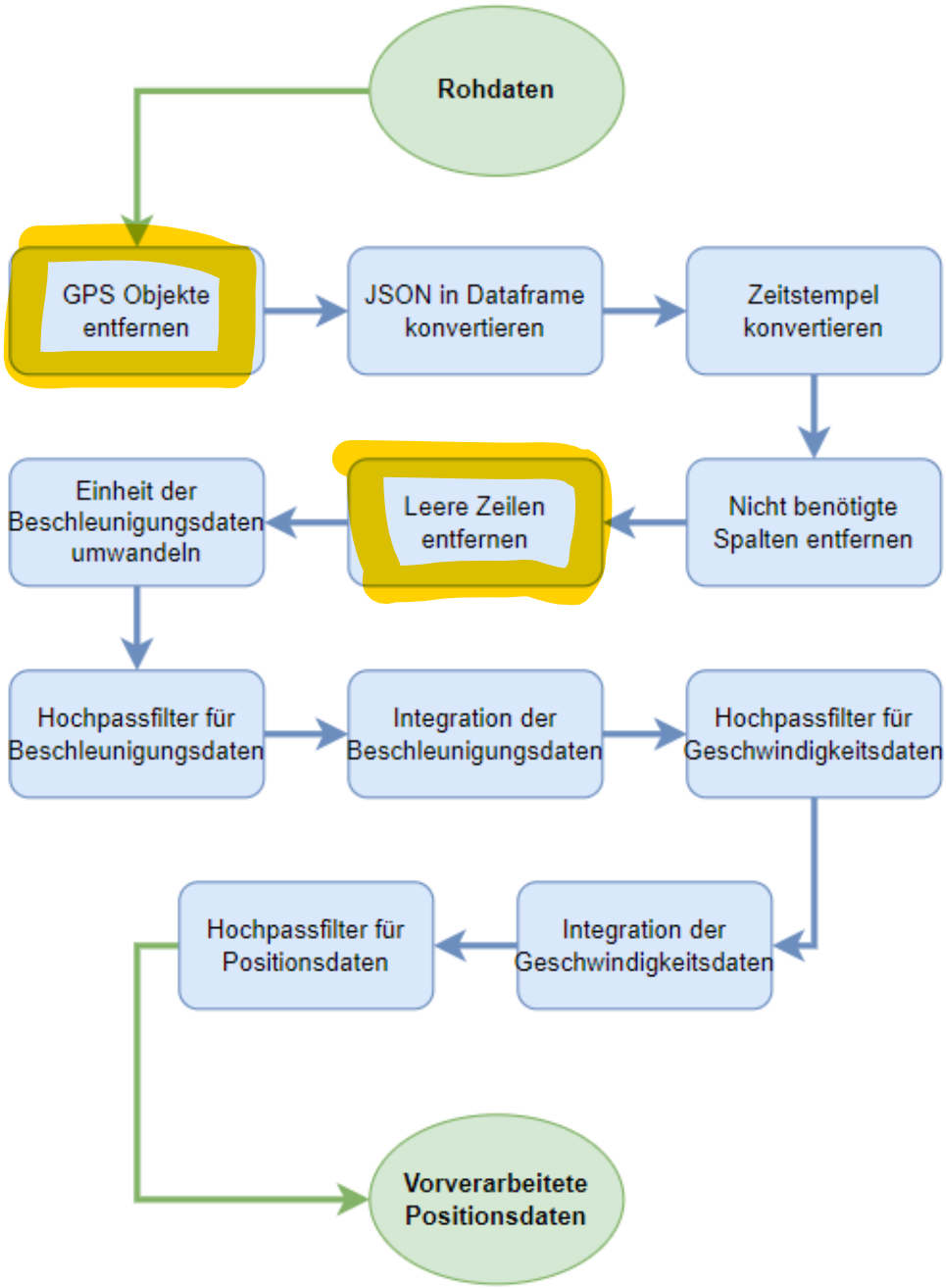

Abbildung 6-1 Datenpipeline

GPS-Objekte entfernen

Im Kapitel 4 wurde bereits erläutert, dass die Rohdaten, welche durch die Sensorbox aufgenommen werden, IMU Objekte und GPS-Objekte enthalten. Da die GPS-Objekte lediglich zur Kalibrierung dienen und nicht zur weiteren Analyse gebraucht werden, werden diese Objekte aus der JSON-Datei entfernt. Somit liegen in der JSON-Datei nur noch IMU-Datei vor.

JSON in Dataframe konvertieren

Zur weiteren Verarbeitung der IMU-Objekte ist es vorteilhaft diese in ein Dataframeformat zu konvertieren. Ein Dataframe ist eine tabellenförmige Datenstruktur, welche eine vereinfachte und beschleunigte Weiterverarbeitung der Dateneinträge ermöglicht. Eine detaillierte Dokumentation der Dataframe-Datenstruktur kann unter folgendem Link gefunden werden:

https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html

Zeitstempel konvertieren

Der Zeitstempel der einzelnen Dateneinträge liegt im Unix-Epoch Format vor. Dabei werden die Sekunden seit dem 01. Januar 1970 gezählt. Um die spätere Interpretation der Analyseergebnisse zu erleichtern und zugänglicher zu machen, bietet es sich an diese Zeitstempel in ein gewöhnliches Datetime-Format umzuwandeln. Eine detaillierte Dokumentation der Dataframe-Datenstruktur kann unter folgendem Link gefunden werden:

https://docs.python.org/3/library/datetime.html

Nicht benötigte Spalten entfernen

Der Abbildung 5-2 können die verschiedenen Parameter entnommen werden, welche von der Sensorbox gemessen werden. Im Kontext der gegebenen Wissenschaftlichen Fragestellung, bei der die Position in X und Y Richtung prognostiziert werden sollen, sind nur die Beschleunigungsdaten in X und Y Richtung relevant. Deshalb können die restlichen Spalten entfernt werden, da diese bei der weiteren Verarbeitung nicht mehr benötigt werden.

Leere Zeilen entfernen

Aus unterschiedlichsten Gründen kann es dazu kommen, dass einige Dateneinträge leer sind. Diese leeren Einträge können bewirken, dass die weitere Analyse der Datensätze verfälscht wird. Deshalb sollten diese Dateneinträge im Rahmen der Datenpipeline entfernt werden. Es gibt zwei verschiedene Methoden dies durchzuführen, die folgend aufgelistet sind:

1. Leere Dateneinträge komplett entfernen.
2. Leere Dateneinträge mit Interpolationswerten befüllen.

Im Rahmen dieser Abschlussarbeit wird die zweite Methode verwendet. Somit werden leere Dateneinträge mit Interpolationswerten befüllt und es kann sichergestellt werden, dass jedem Zeitstempel Messdaten zugeordnet sind. Dies ist für die weitere Verarbeitung der Daten notwendig, da bei einigen Schritten von einer gegebenen Frequenz der Messdaten ausgegangen werden muss.

Einheit der Beschleunigungsdaten umwandeln

Die Beschleunigungsdaten, welche von der Sensorbox erfasst werden, liegen in der Einheit *g* vor (also 9.81 m/s²). Um die spätere Interpretation der Analyseergebnisse zu erleichtern und zugänglicher zu machen, bietet es sich an die Einheit in m/s² umzuwandeln.

Hochpassfilter für Beschleunigungsdaten

Es wird ein Butterworth Hochpassfilter mit einer Grenzfrequenz von 0,5 Hz, einer Ordnung von 11 und einem Padding von 25 s verwendet, um transiente Effekte zu beseitigen und den Mittelwert des Signals auf 0 zu setzen [1][2][5]. Die Mittelwertanpassung ist notwendig, um die Integration im nächsten Schritt nicht zu verfälschen. Der Hochpassfilter wurde ebenfalls im Rahmen verschiedener Studien für die Verarbeitung von ähnlichen Datensätzen verwendet [1][2][5].

Integration der Beschleunigungsdaten

Nun liegen die torsionsbefreiten Beschleunigungsdaten mit einem neutralen Mittelwert vor.
Da wie beschrieben im Rahmen dieser Forschungsarbeit die Prognose der Positionsdaten erforscht werden soll, kann der folgende physikalische Zusammenhang zwischen der Beschleunigung, Geschwindigkeit und Strecke genutzt werden, um diese zu berechnen.

$$v(t) = \int a(t)\ dt \tag{6-1}$$

$$s(t) = \int v(t)\ dt \tag{6-2}$$

Zunächst wird die Integration der Beschleunigungsdaten durchgeführt, um die Geschwindigkeitsdaten zu berechnen.

Hochpassfilter für Geschwindigkeitsdaten

Es wird erneut ein Butterworth Hochpassfilter angewandt (diesmal auf die Geschwindigkeitsdaten), um die durch die Integration entstandenen Torsionseffekte zu entfernen und den Mittelwert der Zeitreihe erneut auf 0 zu setzen. Dies entspricht ebenfalls dem Vorgehen aus den erwähnten Studien [1][2][5].

Integration der Geschwindigkeitsdaten

Wie der Formel 6-2 entnommen werden kann, ist eine erneute Integration (diesmal der Geschwindigkeitsdaten) notwendig, um die zu analysierenden Positionsdaten zu berechnen.

Hochpassfilter für Positionsdaten

Dieser letzte optionale Schritt sorgt dafür, dass die Positionsdaten ebenfalls von Torsionseffekten befreit werden und einen Mittelwert von 0 haben. Diese berechneten Positionsdaten werden dann als Eingabe der Prognosemodelle verwendet und können den folgenden Abbildungen 6-2, 6-3 und 6-4 beispielhaft entnommen werden.

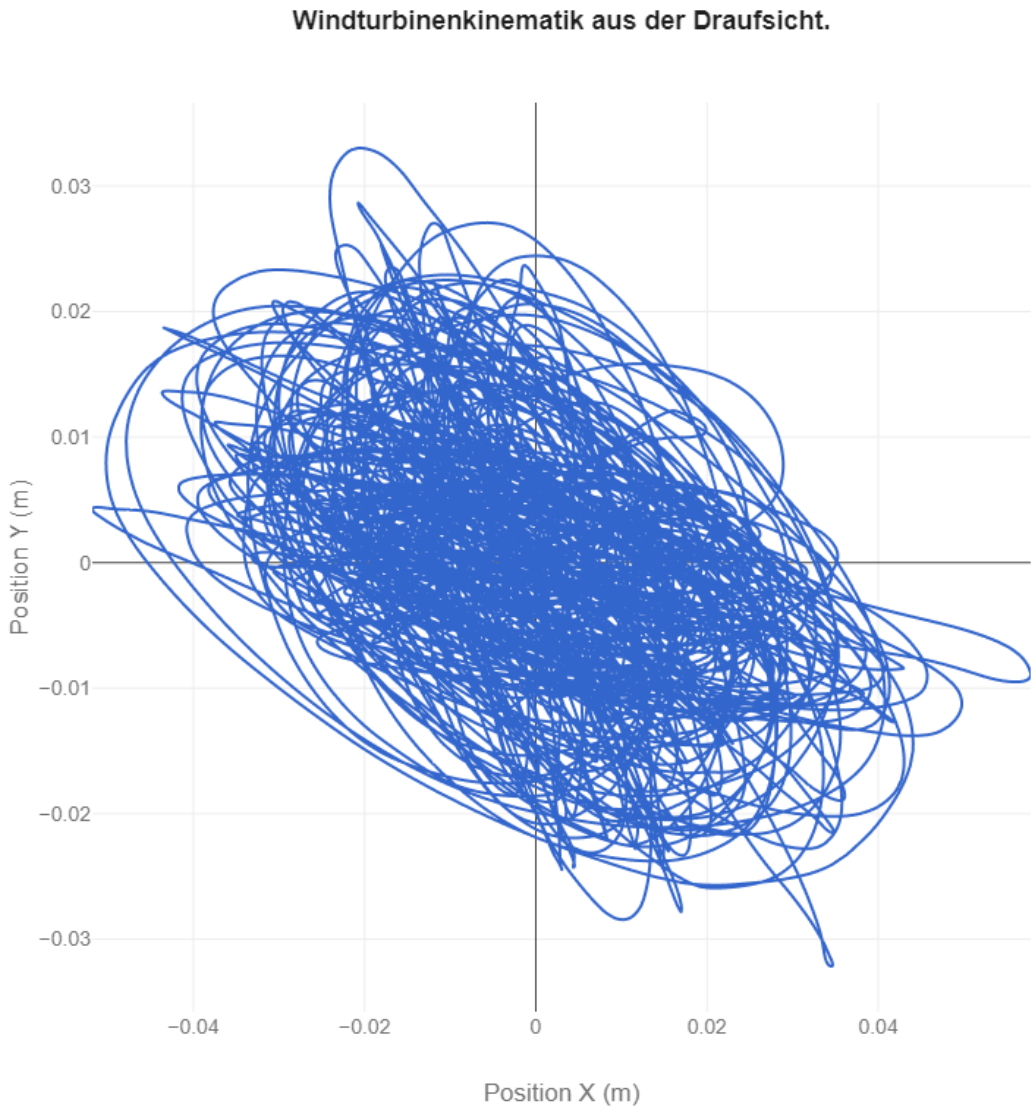

Abbildung 6-2 Windturbinenkinematik aus der Draufsicht

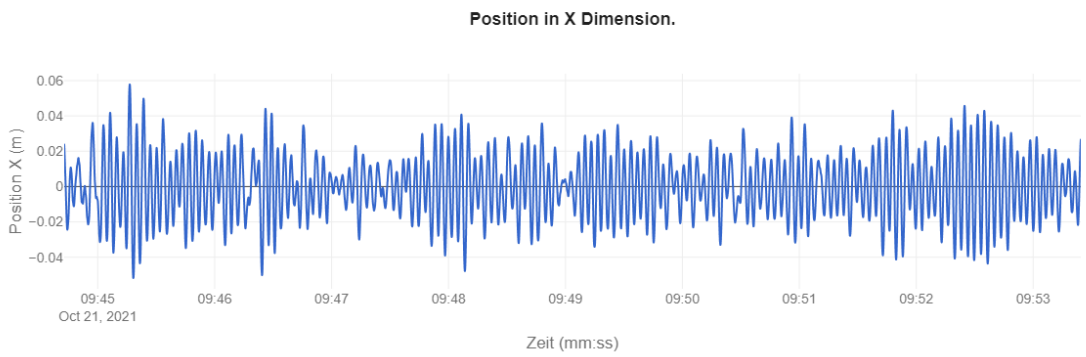

Abbildung 6-3 Positionsdaten in X Dimension

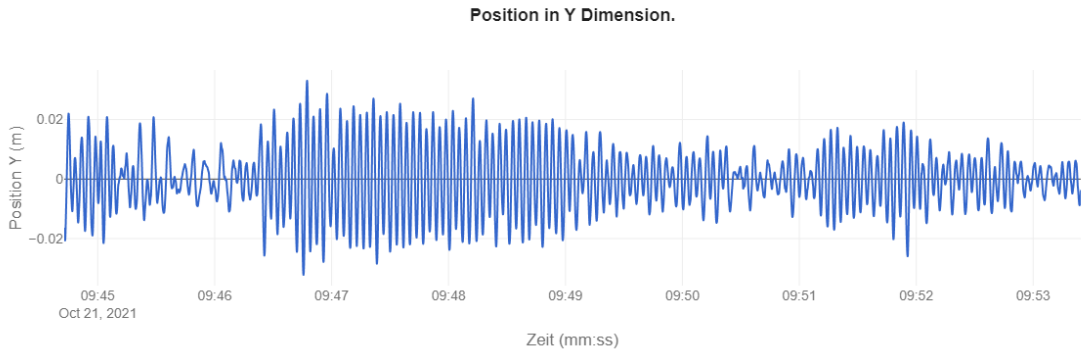

Abbildung 6-4 Positionsdaten in Y Dimension

Dem folgenden Link kann die vorgestellte Datenpipeline als Jupyter Notebook entnommen werden:

https://github.com/WindIO-Bachelorthesis/Shortterm_Forecast/blob/main/data/Preprocessing.ipynb

Im Kapitel 9.3 wird schließlich diese Datenpipeline in Contact Elements for IoT implementiert, sodass ein kontinuierlicher Messdatenstrom verarbeitet werden kann.

7 Kurzzeitprognose der Turmschwingungskinematik

Die theoretischen Grundlagen der Prognosemodelle ARIMA, SARIMA und Prophet wurden bereits im Kapitel 3.2 thematisiert.

Im Folgenden werden für jedes der drei zu implementierenden Prognosemodelle die praktische Umsetzung und die grundlegende Auswertung der Prognosegenauigkeit, für einen Zeitraum von 80 Sekunden, analysiert. Zur Bewertung der Genauigkeit wird der RMSE Wert verwendet, welcher in der einschlägigen Literatur häufig zur Genauigkeitsbestimmung verwendet wird [16][17][18].

Die Berechnung kann der folgenden Formel 7-1 entnommen werden [16][17][18]:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{7-1}$$

Zur Implementierung bzw. zur praktischen Umsetzung der Prognosemodelle wurde die Programmiersprache Python in der Version 3.8 verwendet und folgende Bibliotheken:

Bibliothek	Versionsnummer
pandas	1.2.4
numpy	1.20.1
plotly	5.1.0
statsmodels	0.12.2
scipy	1.6.2
fbprophet	0.7.1

Tabelle 7-1 Bibliotheken zur Prognosemodell Implementierung

7.1 ARIMA

Die theoretischen Grundlagen zum ARIMA Prognosemodell wurden im Kapitel 3.2.1 erläutert. In den folgenden zwei Unterkapiteln werden die praktische Umsetzung des ARIMA Modells und die grundlegende Auswertung der Prognosegenauigkeit für den bereits definierten Zeitraum von 80 Sekunden durchgeführt.

7.1.1 Umsetzung

Die im Kapitel 6 beschriebenen Positionsdaten werden als Eingabe des Prognosemodells verwendet. Dabei wird die Zeitreihe auf eine Frequenz von 1Hz geresampelt. Das bedeutet, dass für jede Sekunde der Durchschnitt über alle zugehörigen Dateneinträge gebildet wird, da nicht die Vorhersage jeder einzelnen Bewegung bei einer Frequenz von 50 Hz von Interesse ist, sondern die Anhäufung bzw. Ansammlung von kinematischen Ereignissen [23]. Danach wird die Zeitreihe in einen Trainigsdatensatz und einen Testdatensatz aufgeteilt. Bei dem Training des Prognosemodells werden die im Kapitel 3.2.1 thematisierten AR, I und MA Parameter automatisch kalibriert, sodass der Prognosefehler minimal ist.

Dem folgenden Link kann die Implementierung des ARIMA-Prognosemodells als Jupyter Notebook entnommen werden:

https://github.com/WindIO-Bachelorthesis/Shortterm_Forecast/blob/main/models/ARIMA.ipynb

Außerdem kann die Implementierung dem folgenden Link als HTML Datei entnommen werden:

https://github.com/WindIO-Bachelorthesis/Shortterm_Forecast/blob/main/models/HTMLExports/ARIMA.html

7.1.2 Auswertung

Den folgenden Abbildung 7-1 und 7-2 kann die Prognose des ARIMA Modells für einen Zeitraum von 80 Sekunden und der zugehörige Testdatensatz entnommen werden.

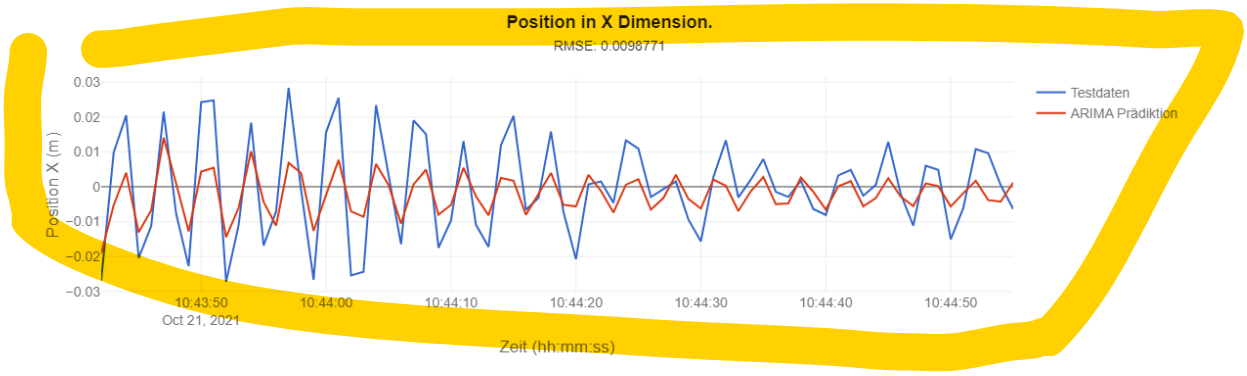

Abbildung 7-1 ARIMA Prognose für Positionsdaten in X Dimension

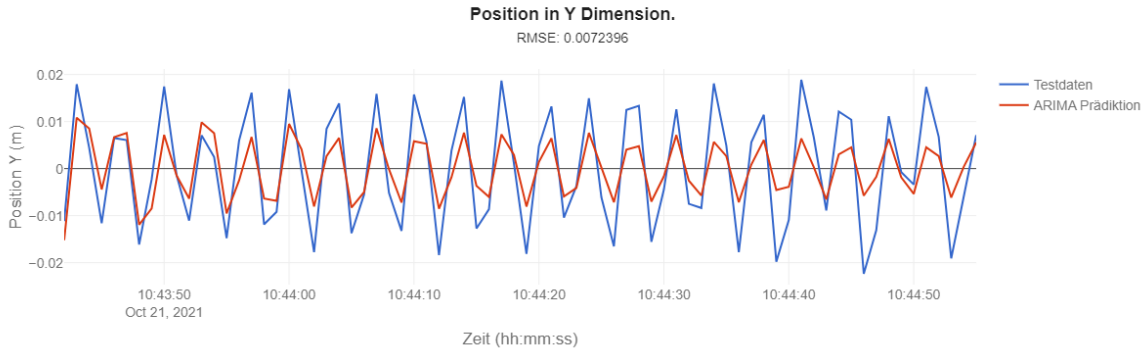

Abbildung 7-2 ARIMA Prognose für Positionsdaten in Y Dimension

Im Kapitel 7.4 werden die RMSE Werte für die Position in X und Y Richtung mit der Prognosegenauigkeit der anderen Modelle verglichen und die Prognose für verschiedene Zeiträume ausgewertet.

7.2 SARIMA

Die theoretischen Grundlagen zum SARIMA Prognosemodell wurden im Kapitel 3.2.2 erläutert. In den folgenden zwei Unterkapiteln werden die praktische Umsetzung des SARIMA Modells und die grundlegende Auswertung der Prognosegenauigkeit für den bereits definierten Zeitraum von 80 Sekunden durchgeführt.

7.2.1 Umsetzung

Die im Kapitel 6 beschriebenen Positionsdaten werden als Eingabe des Prognosemodells verwendet. Dabei wird die Zeitreihe auf eine Frequenz von 1Hz geresampelt. Das bedeutet, dass für jede Sekunde der Durchschnitt über alle zugehörigen Dateneinträge gebildet wird, da nicht die Vorhersage jeder einzelnen Bewegung bei einer Frequenz von 50 Hz von Interesse ist, sondern die Anhäufung bzw. Ansammlung von kinematischen Ereignissen [23]. Danach wird die Zeitreihe in einen Trainigsdatensatz und einen Testdatensatz aufgeteilt. Bei dem Training des Prognosemodells werden die im Kapitel 3.2.2 thematisierten AR, I, MA und saisonalen Parameter automatisch kalibriert, sodass der Prognosefehler minimal ist.

Dem folgenden Link kann die Implementierung des SARIMA-Prognosemodells als Jupyter Notebook entnommen werden:

https://github.com/WindIO-Bachelorthesis/Shortterm_Forecast/blob/main/models/SARIMA.ipynb

Außerdem kann die Implementierung dem folgenden Link als HTML Datei entnommen werden:

https://github.com/WindIO-Bachelorthesis/Shortterm_Forecast/blob/main/models/HTMLExports/SARIMA.html

7.2.2 Auswertung

Den folgenden Abbildung 7-3 und 7-4 kann die Prognose des SARIMA Modells für einen Zeitraum von 80 Sekunden und der zugehörige Testdatensatz entnommen werden.

Abbildung 7-3 SARIMA Prognose für Positionsdaten in X Dimension

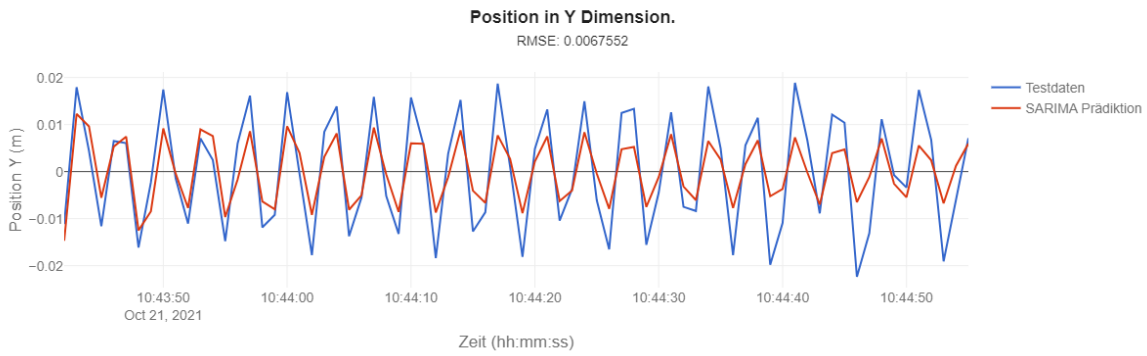

Abbildung 7-4 SARIMA Prognose für Positionsdaten in Y Dimension

Im Kapitel 7.4 werden die RMSE Werte für die Position in X und Y Richtung mit der Prognosegenauigkeit der anderen zwei Modelle verglichen und die Prognose für verschiedene Zeiträume ausgewertet.

7.3 Prophet (Neuronales Netzwerk)

Die theoretischen Grundlagen zum Prophet Prognosemodell wurden im Kapitel 3.2.3 erläutert. In den folgenden zwei Unterkapiteln werden die praktische Umsetzung des Prophet Modells und die grundlegende Auswertung der Prognosegenauigkeit für den bereits definierten Zeitraum von 80 Sekunden durchgeführt.

7.3.1 Umsetzung

Die im Kapitel 6 beschriebenen Positionsdaten werden als Eingabe des Prognosemodells verwendet. Dabei wird die Zeitreihe auf eine Frequenz von 1Hz geresampelt. Das bedeutet, dass für jede Sekunde der Durchschnitt über alle zugehörigen Dateneinträge gebildet wird, da nicht die Vorhersage jeder einzelnen Bewegung bei einer Frequenz von 50 Hz von Interesse ist, sondern die Anhäufung bzw. Ansammlung von kinematischen Ereignissen [23]. Danach wird die Zeitreihe in einen Trainigsdatensatz und einen Testdatensatz aufgeteilt. Bei dem Training des Prognosemodells werden die im Kapitel 3.2.2 thematisierten Teilfunktionen werden automatisch kalibriert, sodass der Prognosefehler minimal ist.

Dem folgenden Link kann die Implementierung des Prophet-Prognosemodells als Jupyter Notebook entnommen werden:

https://github.com/WindIO-Bachelorthesis/Shortterm_Forecast/blob/main/models/PROPHET.ipynb

Außerdem kann die Implementierung den folgendem Link als HTML Datei entnommen werden:

https://github.com/WindIO-Bachelorthesis/Shortterm_Forecast/blob/main/models/HTMLExports/PROPHET.html

7.3.2 Auswertung

Den folgenden Abbildung 7-5 und 7-6 kann die Prognose des Prophet Modells für einen Zeitraum von 80 Sekunden und der zugehörige Testdatensatz entnommen werden.

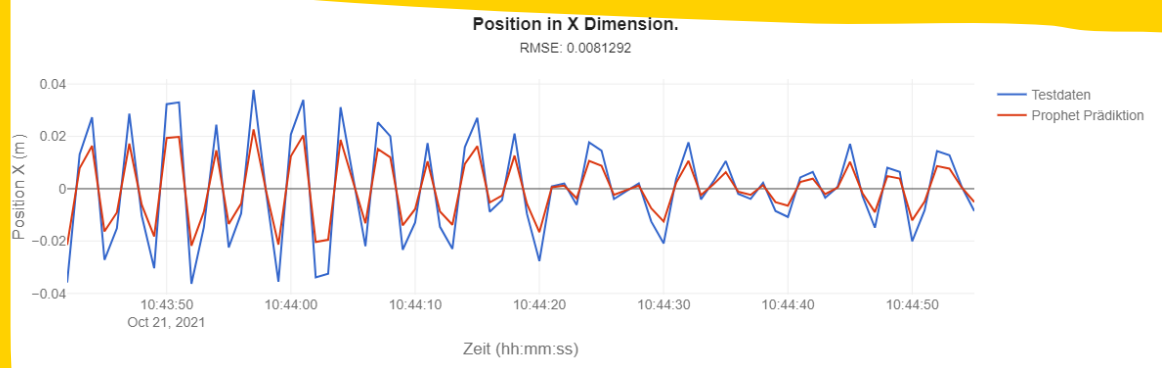

Abbildung 7-5 Prophet Prognose für Positionsdaten in X Dimension

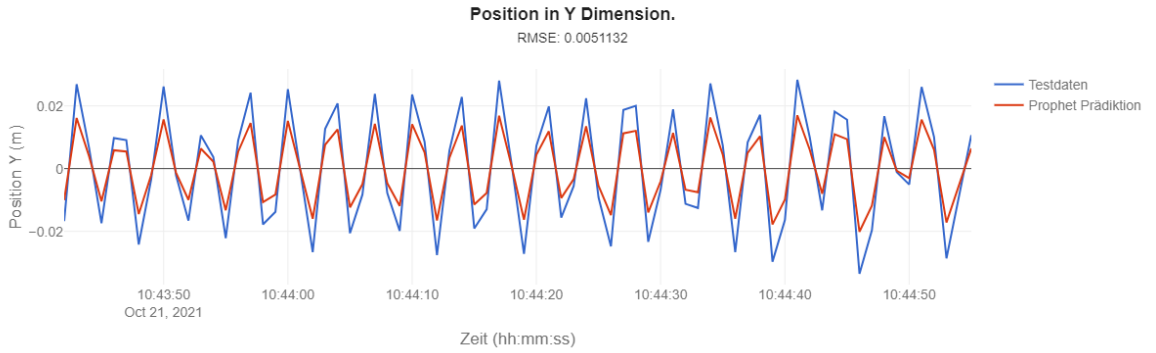

Abbildung 7-6 Prophet Prognose für Positionsdaten in Y Dimension

Im Kapitel 7.4 werden die RMSE Werte für die Position in X und Y Richtung mit der Prognosegenauigkeit der anderen zwei Modelle verglichen und die Prognose für verschiedene Zeiträume ausgewertet.

7.4 Vergleich der Prognosemodellergebnisse

Der folgenden Tabelle 7-4 können die RMSE Werte für die Abbildungen 7-1, 7-2, 7-3, 7-4, 7-5 und 7-6 entnommen werden.

Prognosemodell	Position in X Dimension	Position in Y Dimension
ARIMA	0.0098771	0.0072396
SARIMA	0.0095985	0.0067552
Prophet	0.0081292	0.0051132

Tabelle 7-2 Bibliotheken zur Prophet Implementierung

Der Tabelle ist zu entnehmen, dass das SARIMA Modell (für den Zeitraum vom 80 Sekunden) ein geringfügig besseres Prognoseergebnis als das ARIMA Modell bereitstellt und das Prophet Neuronale Netz noch genauer ist. Die Abbildung 7-7 und 7-8 zeigen die Prognosegenauigkeit für verschiedene Zeiträume. Somit kann die wissenschaftliche Fragestellung, wie gut die Turmkinematik prognostiziert werden kann, fundiert beantwortet werden und die thematisierten Modelle verglichen werden.

Zur Berechnung der RMSE-Werte werden 15 verschiedene Zeiträume definiert, für welche einzeln die Genauigkeit analysiert bzw. berechnet wird. Diese werden dann im Diagramm abgebildet. Um die Prognosegenauigkeit und den RMSE Wert im Kontext beurteilen zu können, wird ein Schwellwert benötigt, welcher in der einschlägigen Literatur als Baseline bezeichnet wird. Da die simpelste Form der Prognose der Mittelwert der Zeitreihe ist, kann die Standardabweichung als geeignete Baseline verwendet werden. Die Berechnung kann der folgenden Formel 7-2 entnommen werden:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \mu)^2} \tag{7-2}$$

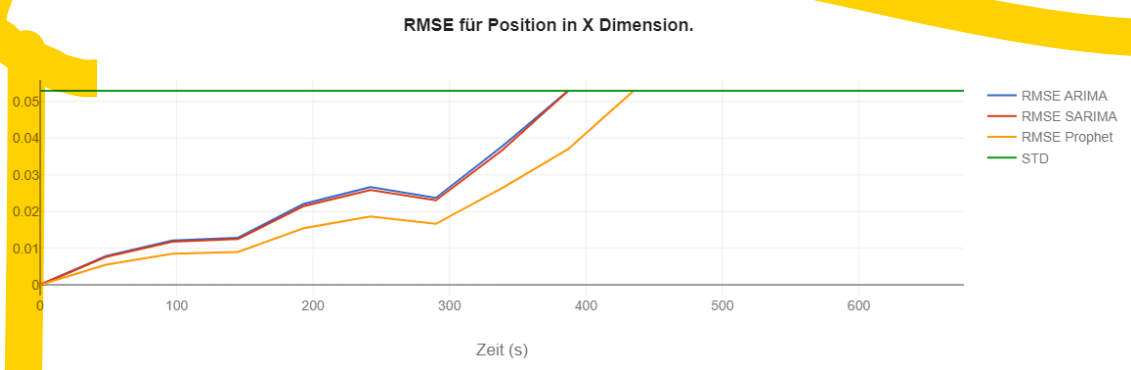

Abbildung 7-7 Prognosegenauigkeit der verschiedenen Modelle für Position in X Dimension

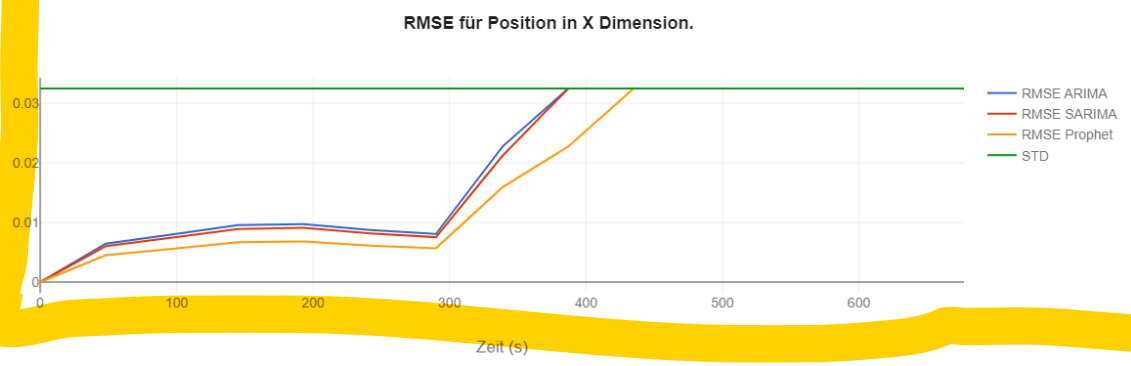

Abbildung 7-8 Prognosegenauigkeit der verschiedenen Modelle für Position in Y Dimension

Die Abbildungen zeigen, dass alle drei Prognosemodelle für die Position in X Dimension eine gute Prognose für einen Zeitraum von 180 Sekunden liefern und für die Position in Y Dimension eine gute Prognose für einen Zeitraum von bis zu 290 Sekunden liefern. Somit kann der Vergleich auf Grundlage von Tabelle 7-4 bestätigt werden.

Um die Modelle jedoch vollständig zu bewerten und charakterisieren, muss neben der Prognosegenauigkeit auch die notwendige Rechenzeit mit einbezogen werden. Dabei ist besonders hervorzuheben, dass das SARIMA Modell aufgrund der zusätzlichen saisonalen Parameter deutlich mehr Rechenkapazität benötigt als das ARIMA Modell. Wenn man das vor dem Hintergrund der nur leicht besseren Prognosegenauigkeit betrachtet, kann gesagt werden, dass das SARIMA Modell zur praktischen Verwendung ungeeignet ist. Das ARIMA und Prophet Modell sind aufgrund der guten Prognoseergebnisse und der geringen Rechenzeit besonders für IoT Szenarien geeignet. Die Implementierung des ARIMA und Prophet Modells in Contact Elements for IoT wird im Kapitel 9.5 thematisiert.

8 Optimierung der Prognosemodelle durch Fourier Transformation

In diesem Kapitel sollen die Prognosemodelle in der Genauigkeit ihrer Ergebnisse, mithilfe der Fourier Transformation, optimiert werden. Die folgenden vier Unterkapitel behandeln die theoretische Grundlage des Optimierungsansatzes, die praktische Umsetzung, die Auswertung der Ergebnisse und die Integration dieses zusätzliches Arbeitsschritts in die Datenpipeline, welche im Kapitel 6 thematisiert wurde.

8.1 Theoretische Grundlage

Der folgenden Abbildung 8-1 kann beispielhaft entnommen werden, dass unsere Positionsdaten aus der additiven Überlagerung einer Grundfunktion und einem White-Noise besteht. Bei dem Training der Prognosemodelle durch diese insgesamt rauschende Funktion, kommt es zu einem Overfitting des Modells. Unter Overfitting versteht man, dass das Prognosemodell im Trainingsvorgang zu stark auf das Rauschen trainiert wird und weniger auf die zugrundeliegende Funktion, welche rot markiert ist [7][8]. Indem man das Rauschen entfernt und lediglich die zugrundeliegende Funktion zum Training des Prognosemodells verwendet, kann ein Overfitting vermieden werden und dies kann zu besseren Prognoseergebissen (selbst für rauschende Testdaten) führen [7][8].

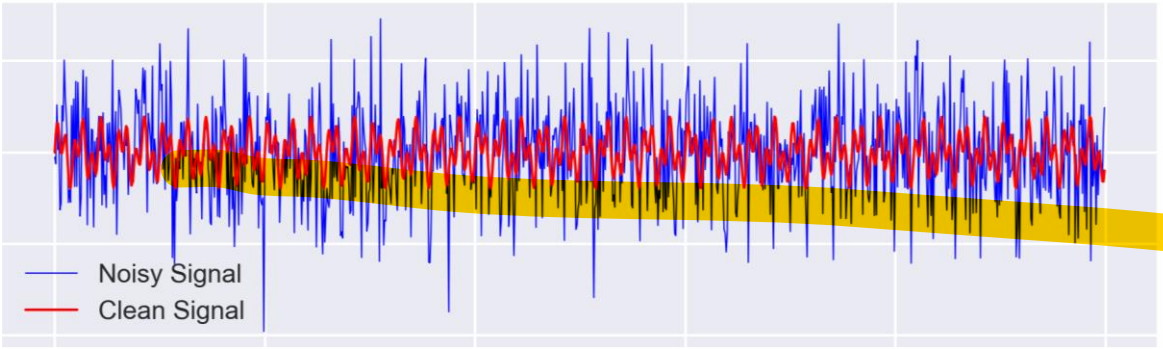

Abbildung 8-1 Beispielhafte additive Signalüberlagerung

8.2 Umsetzung

Die im Kapitel 6 beschriebenen Positionsdaten werden als Eingabe des Prognosemodells verwendet. Dabei wird die Zeitreihe auf eine Frequenz von 1Hz geresampelt. Das bedeutet, dass für jede Sekunde der Durchschnitt über alle zugehörigen Dateneinträge gebildet wird, da nicht die Vorhersage jeder einzelnen Bewegung bei einer Frequenz von 50 Hz von Interesse ist, sondern die Anhäufung bzw. Ansammlung von kinematischen Ereignissen [23]. Danach wird die Zeitreihe in einen Trainingsdatensatz und einen Testdatensatz aufgeteilt. Folgend wird, wie im Kapitel 8.1 erläutert der Trainingsdatensatz vom Rauschen befreit und der Testdatensatz wird nicht weitergehend verarbeitet. Zur Implementierung wurden folgende Bibliotheken verwendet:

Bibliothek	Versionsnummer
pandas	1.2.4
numpy	1.20.1
plotly	5.1.0
scipy	1.6.2
matplotlib	3.3.4

Tabelle 8-1 Bibliotheken zur Prophet Implementierung

Zur Entfernung des Signalrauschens bzw. zum Entfernen des White Noise der Trainingsdaten sind konkret drei Schritte notwendig:

1. Fourier Transformation.
2. Niedrige Amplitudenwerte gleich 0 setzen.
3. Rücktransformation.

Dieses Vorgehen kann ebenfalls der folgenden Abbildung 8-2 entnommen werden.

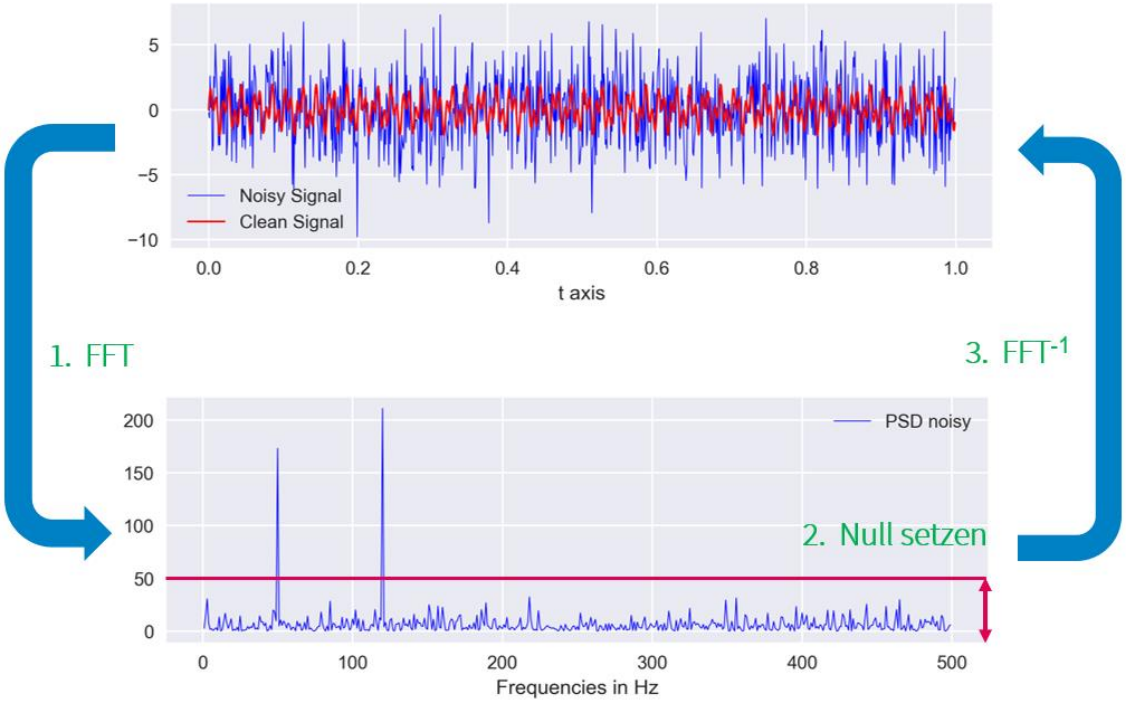

Abbildung 8-2 Beispielhafte Fourier Transformation zur Rauschentfernung

Andere Studien haben Fourier Transformationen im Zusammenhang mit verwandten Datensätzen verwendet, um die Kinematik der Windenergieanlagen beschrieben zu können [1][2][24][25]. Dabei wurden die Beschleunigungsdaten der Messungen analysiert. Diese haben festgestellt, dass das Frequenzspektrum zwischen 0.2 und 0.3 Hz ein Maxima aufweist, welches als Eigenfrequenz der Turmschwingungskinematik interpretiert werden kann [1][2][24][25]. In der folgenden Abbildung ist das Frequenzspektrum der Beschleunigungsdaten von der Senvion-Anlage abgebildet.

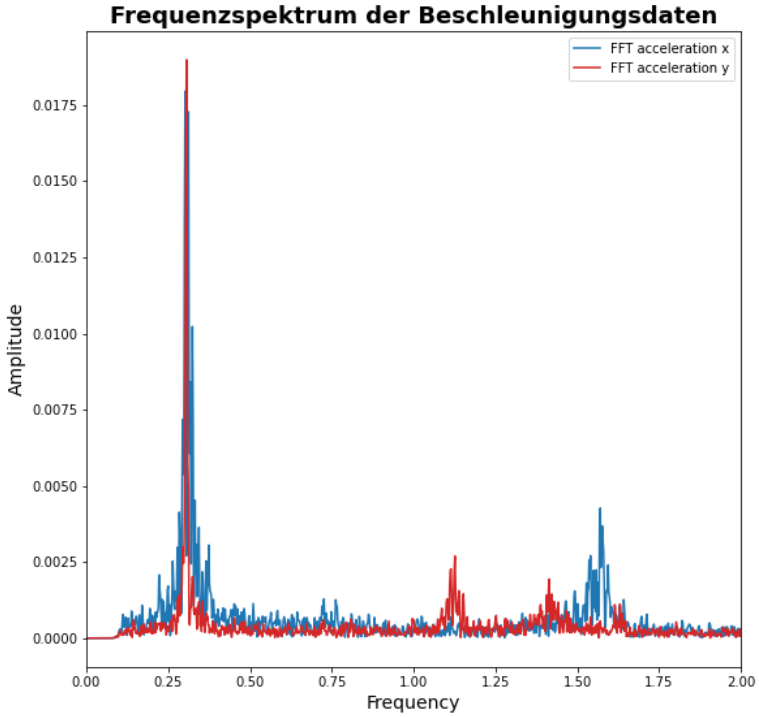

Abbildung 8-2 Frequenzspektrum der Beschleunigungsdaten der Senvion-Anlage

Somit bestätigt dieses Frequenzspektrum der Beschleunigungsdaten die dominante Eigenfrequenz bei ca. 0.27 Hz.

Jedoch sollen wie bereits beschrieben, im Rahmen dieser Abschlussarbeit die Positionsdaten analysiert werden, weshalb der folgenden Abbildung das Frequenzspektrum der Positionsdaten entnommen werden kann.

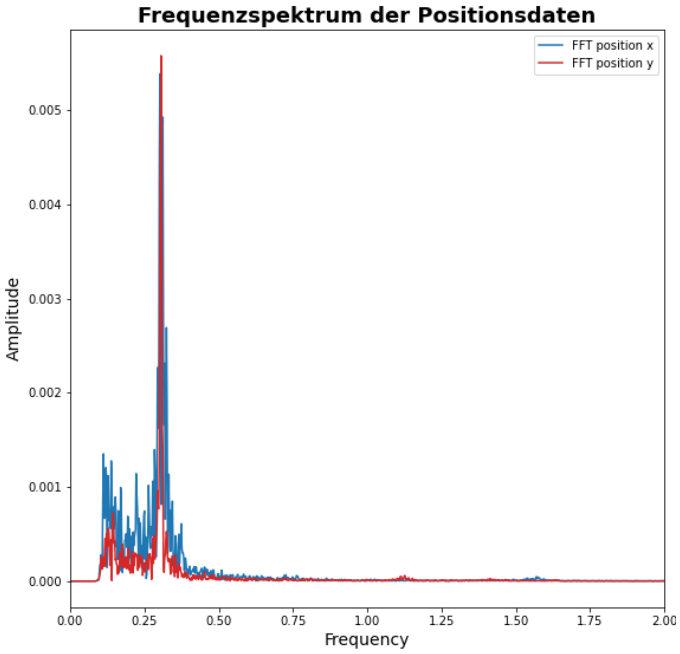

Abbildung 8-3 Frequenzspektrum der Positionsdaten der Senvion-Anlage

Auch bei den Positionsdaten ist ein Maxima bei ca. 0.27 Hz zu erkennen.

Welche Werte werden jetzt alle auf 0 gesetzt?

Was ist der Schwellwert?

Wie sieht das Rücktransformierte Testdatensignal aus?

8.3 Auswertung

RMSE Abbildung wie bei Abbildung 7-7 und 7-8.

Sind die Prognose Ergebnisse durch den Optimierungsansatz besser?

Wieso sind diese Eventuell doch nicht besser?

8.4 Integration in die Datenpipeline

Wie kann man die Fourier Transformation in die Datenpipeline integrieren, um die Prognose Ergebnisse zu verbessern.

9 Contact Elements Integration

Contact Elements for IoT ist die IoT Lösung von Contact Software. Die Abkürzung IoT steht für „Internet of Things“ und beschreibt die Digitale Vernetzung von Assets über das Internet. Durch diese kontinuierliche Ansammlung von Sensordaten, ist es möglich Prozesse zu optimieren, Kosten zu sparen und strategische Entscheidungen auf Grundlage dieser Datenbasis zu treffen. Contact Elements for IoT wird im Rahmen des WindIO Forschungsprojekts zur Erstellung eines kommerziellen Digitalen Zwillings verwendet.
Aktuell arbeitet das Unternehmen an der Entwicklung eines Datenanalysemodus dessen grundlegende Funktionalität, für den WindIO-Projektkontext, der folgenden schematischen Darstellung entnommen werden kann.

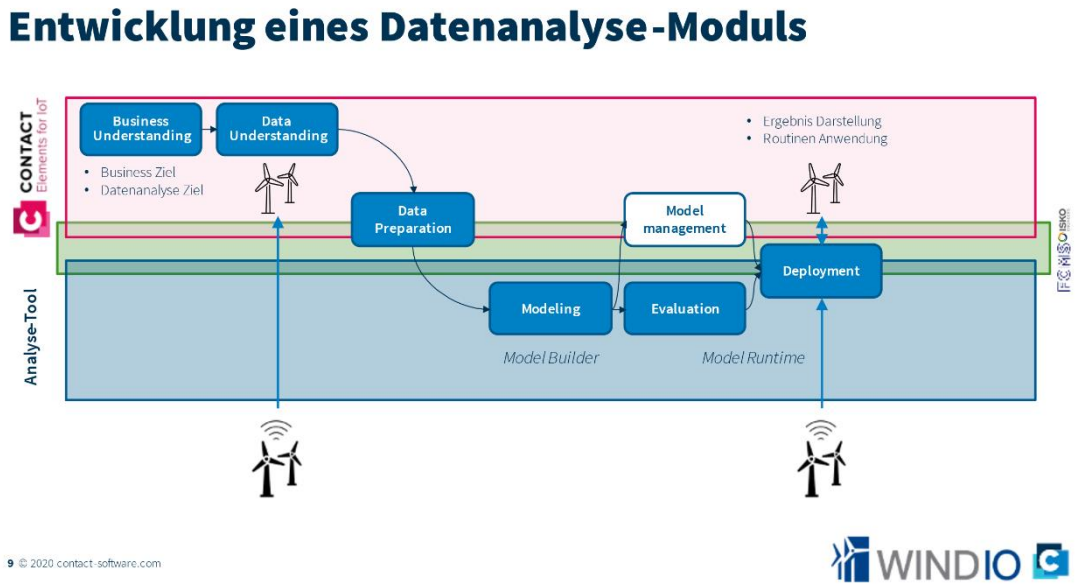

Abbildung 9-1 Contact Elements for IoT Entwicklung eines Datenanalyse-Moduls

Die Integration der Prognosemodelle und die Implementierung der Datenpipeline, lassen sich in die thematischen Blöcke „Data Preparation“ und „Modeling“ einordnen. In den folgenden Unterkapiteln werden die grundlegende Konfiguration des Assets, die Möglichkeit der Job- und Vorlagenautomatisierung, die Implementierung der Datenpipeline, die Visualisierung der Positionsdaten, die Echtzeitprognose der Turmschwingungskinematik und die Erweiterungsmöglichkeiten der Implementierung thematisiert.

9.1 Konfiguration des Assets

Wie ist das Asset konfiguriert?

9.2 Automatisierung Jobs und Vorgänge

Was ist die Automatisierung und in welchem Rahmen wurde diese entwickelt?

Welche Möglichkeiten hat man hier eine Automatisierung zu gestalten?

Was sind Vorgänge und wie kann man sie praktisch für die gegebene Fragestellung verwenden?

9.3 Implementierung der Datenpipeline

Wie wird Kapitel 6 in CE4IoT umgesetzt?

9.4 Datenvisualisierung

Welche Standardfunktionalität gibt es?

Warum reicht das eventuell nicht aus und wie kann man diese Möglichkeiten erweitern?

9.5 Echtzeitprognose der Turmschwingungskinematik

Auf Kapitel 8.2 eingehen und aufzeigen, wie das ARIMA Modell mit der zur Verfügung stehenden Funktionalität umgesetzt wurde?

9.6 Erweiterungsmöglichkeiten

Wie kann man im IoT Kontext diese Prognose verbessern bzw. optimieren?

Wie kann man die Lösung auf andere Fragestellungen möglichst Ressourcenschonend adaptieren?

10 Fazit

Ziel und Vorgehensweise der Abschlussarbeit zusammenfassen.

Ergebnisse erläutern und zusammenfassen.

Abschließende Beantwortung der Forschungsfrage.

11 Literatur

[1] A. Sander *et al.,* „Relative Motion During Single Blade Installation: Measurements From the North Sea“ in *ASME 2020 39th International Conference on Ocean, Offshore and Arctic Engineering*, Virtual, Online, 2020.

[2] A. Sander, C. Meinhardt und K.-D. Thoben, „MONITORING OF OFFSHORE WIND TURBINES UNDER WAVE AND WIND LOADING DURING INSTALLATION“ in *XI International Conference on Structural Dynamics*, Athens, Greece, 2020, S. 2189–2205, doi: 10.47964/1120.9178.19731.

[3] C. Sun und V. Jahangiri, „Fatigue damage mitigation of offshore wind turbines under real wind and wave conditions“, *Engineering Structures*, Jg. 178, S. 472–483, 2019, doi: 10.1016/j.engstruct.2018.10.053.

[4] International Renewable Energy Agency, „Renewable power generation costs in 2019“.

[5] A. Sander, B. Holman und A. Haselsteiner, „Could mass eccentricity explain the formation of orbits in wind turbines?“, 25. Okt. 2021. [Online]. Verfügbar unter: http://arxiv.org/pdf/2110.12802v1.

[6] A. S. Verma, Z. Jiang, Z. Ren, Z. Gao und N. P. Vedvik, „Response-Based Assessment of Operational Limits for Mating Blades on Monopile-Type Offshore Wind Turbines“, *Energies*, Jg. 12, Nr. 10, S. 1867, 2019, doi: 10.3390/en12101867.

[7] K. Neusser, *Zeitreihenanalyse in den Wirtschaftswissenschaften,* 3. Aufl. Wiesbaden: Vieweg + Teubner, 2011.

[8] R. Schlittgen, *Angewandte Zeitreihenanalyse Mit R,* 3. Aufl. Berlin/München/Boston: Walter de Gruyter GmbH, 2015. [Online]. Verfügbar unter: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=1867274

[9] A. A. Ariyo, A. O. Adewumi und C. K. Ayo, „Stock Price Prediction Using the ARIMA Model“ in *2014 UKSim-AMSS 16th International Conference on*

Modelling and Simulation (UKSim), Cambridge, United Kingdom, 2014, S. 106–112, doi: 10.1109/UKSim.2014.67.

[10] S. Siami-Namini, N. Tavakoli und A. Siami Namin, „A Comparison of ARIMA and LSTM in Forecasting Time Series“ in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Orlando, FL, 2018, S. 1394–1401, doi: 10.1109/ICMLA.2018.00227.

[11] G. Zhang, „Time series forecasting using a hybrid ARIMA and neural network model“, *Neurocomputing*, Jg. 50, S. 159–175, 2003, doi: 10.1016/S0925-2312(01)00702-0.

[12] J. Vogel, „ARIMA- und SARIMA-Modelle“ in *Prognose von Zeitreihen*, J. Vogel, Hg., Wiesbaden: Springer Fachmedien Wiesbaden, 2015, S. 123–143, doi: 10.1007/978-3-658-06837-0_6.

[13] K. Kalpakis, D. Gada und V. Puttagunta, „Distance measures for effective clustering of ARIMA time-series“ in *2001 IEEE International Conference on Data Mining*, San Jose, CA, USA, 2001, S. 273–280, doi: 10.1109/ICDM.2001.989529.

[14] K.-Y. Chen und C.-H. Wang, „A hybrid SARIMA and support vector machines in forecasting the production values of the machinery industry in Taiwan“, *Expert Systems with Applications*, Jg. 32, Nr. 1, S. 254–264, 2007, doi: 10.1016/j.eswa.2005.11.027.

[15] T. Fang und R. Lahdelma, „Evaluation of a multiple linear regression model and SARIMA model in forecasting heat demand for district heating system“, *Applied Energy*, Jg. 179, S. 544–552, 2016, doi: 10.1016/j.apenergy.2016.06.133.

[16] S. Liu, Y. Jiao, Q. Sun und J. Jiang, „Estimation of Sea Level Change in the South China Sea from Satellite Altimetry Data“, *Scientific Programming*, Jg. 2021, S. 1–7, 2021, doi: 10.1155/2021/6618135.

[17] F. F. Nobre, A. B. Monteiro, P. R. Telles und G. D. Williamson, „Dynamic linear model and SARIMA: a comparison of their forecasting performance in epidemiology“ (eng), *Statistics in medicine*, Jg. 20, Nr. 20, S. 3051–3069, 2001.

[18] R. W. Divisekara, G. J. M. S. R. Jayasinghe und K. W. S. N. Kumari, „Forecasting the red lentils commodity market price using SARIMA models“, *SN Bus Econ*, Jg. 1, Nr. 1, 2021, doi: 10.1007/s43546-020-00020-x.

[19] M. Daraghmeh, A. Agarwal, R. Manzano und M. Zaman, „Time Series Forecasting using Facebook Prophet for Cloud Resource Management“ in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, Montreal, QC, Canada, 2021, S. 1–6, doi: 10.1109/ICCWorkshops50388.2021.9473607.

[20] S. Liu, Y. Jiao, Q. Sun und J. Jiang, „Estimation of Sea Level Change in the South China Sea from Satellite Altimetry Data“, *Scientific Programming*, Jg. 2021, S. 1–7, 2021, doi: 10.1155/2021/6618135.

[21] S. J. Taylor und B. Letham, *Forecasting at scale*, 2017.

[22] T. Toharudin, R. S. Pontoh, R. E. Caraka, S. Zahroh, Y. Lee und R. C. Chen, „Employing long short-term memory and Facebook prophet model in air temperature forecasting“, *Communications in Statistics - Simulation and Computation*, S. 1–24, 2021, doi: 10.1080/03610918.2020.1854302.

[23] P. Naaijen, K. van Oosten, K. Roozen und R. van 't Veer, „Validation of a Deterministic Wave and Ship Motion Prediction System“ in *ASME 2018 37th International Conference on Ocean, Offshore and Arctic Engineering*, Madrid, Spain, 2018, doi: 10.1115/OMAE2018-78037.

[24] C. Meinhardt, „Application of a 240 Metric Ton Dual-Use Tuned Mass Damper System“ in *Lecture Notes in Civil Engineering, Experimental Vibration Analysis for Civil Structures*, J. P. Conte et al., Hg., Cham: Springer International Publishing, 2018, S. 536–546, doi: 10.1007/978-3-319-67443-8_46.

[25] A. S. Verma, Z. Jiang, Z. Ren, Z. Gao und N. P. Vedvik, „Response-Based Assessment of Operational Limits for Mating Blades on Monopile-Type Offshore Wind Turbines“, *Energies*, Jg. 12, Nr. 10, S. 1867, 2019, doi: 10.3390/en12101867.