Bachelor-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

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt habe. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht.

Bremen, den 24. März 2022 _____

Zelgai, Nemati

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
$\bar{\mu}$	Summe gewichteter Durchschnittswerte
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 Gewichtungsfunktion
g	Erdbeschleunigung
$a(t)$	Beschleunigung
$v(t)$	Geschwindigkeit
$s(t)$	Strecke
$\hat{y}_i$	Tatsächlicher Wert 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 [1-3]. 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 [1-4].

Heutzutage werden die meisten Windenergieanlagen komponentenweise installiert und um die Kosten weiterhin zu senken, ist es insbesondere notwendig, den Installationsprozess zu optimieren [1-3]. 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.

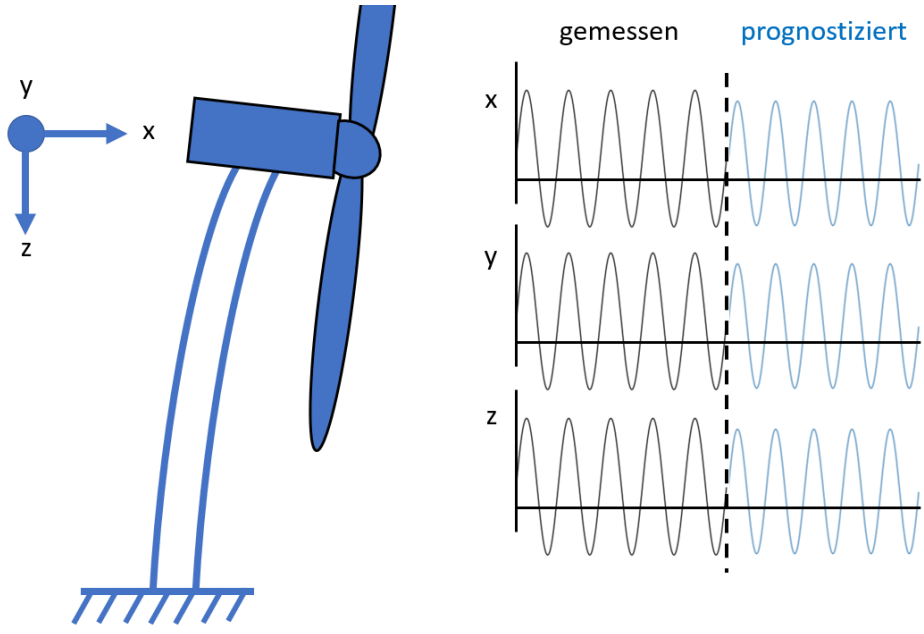

Abbildung 2-1 Prognose der Turmschwingungskinematik.

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 werden die zu analysierenden Daten nicht auf Stationarität überprüft, da diese Angabe nicht von den Prognosemodellen ARIMA, SARIMA und Prophet benötigt wird [7-12].

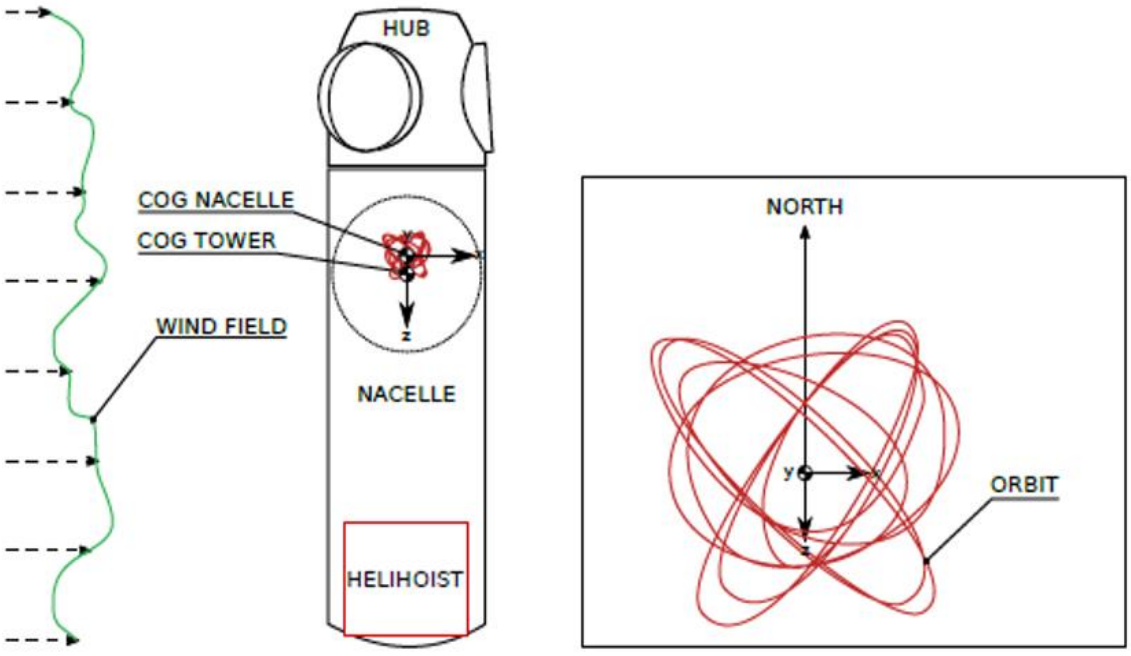

Abbildung 3-1 Auswertung der Installationsdaten vom Offshore Windpark: „Trianel Windpark Borkum 2“ [5].
COG bezeichnet dabei den Massenmittelpunkt eines Körpers.

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. Diese zufälligen Störgrößen können zu einer zeitlich verschiedenen Charakteristiken des Schwingungssignals führen. Deshalb sollte ein gewähltes Prognosemodell dazu in der Lage sein sich diesbezüglich zeitlich verändernde Eigenschaften des Schwingungssignals berücksichtigen zu können.

Kriterium 2

Im Kapitel 3.1 wurde erläutert, dass die Stationarität der zu analysierenden Zeitreihen nicht bewertet wird. Deshalb sollte ein gewähltes Prognosemodell auch keine speziellen Anforderungen an die Stationarität der Zeitreihendaten 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 präzise Ergebnisse für die Strukturkinematikprognose liefern.

In den folgenden fünf Unterkapiteln werden das ARIMA, SARIMA und Prophet-Prognosemodell theoretisch eingeführt. Kapitel 3.2.1 und 3.2.2 beziehen sich dabei auf Subkomponenten des ARIMA und SARIMA-Prognosemodelltyps.

3.2.1 Autoregression (AR)

Autoregressive-Prognosemodelle, abgekürzt AR-Prognosemodelle, stellen die einfachste Schätzmethode dar und können nur auf stationäre Datensätze angewandt werden [13 (Seite 23-26)][14 (Seite 61)]. AR-Modelle prognostizieren zukünftige Parameterwerte, anhand der vergangenen Werte, welche im Fachjargon bzw. in der einschlägigen Literatur auch als „Lags“ bezeichnet werden [13 (Seite 21)][14 (Seite 11)]. Diese Abhängigkeit zu vergangenen Werten (bzw. der Einfluss der vergangenen auf zukünftige Werte) kann der nachfolgenden Abbildung 3-2 entnommen werden.

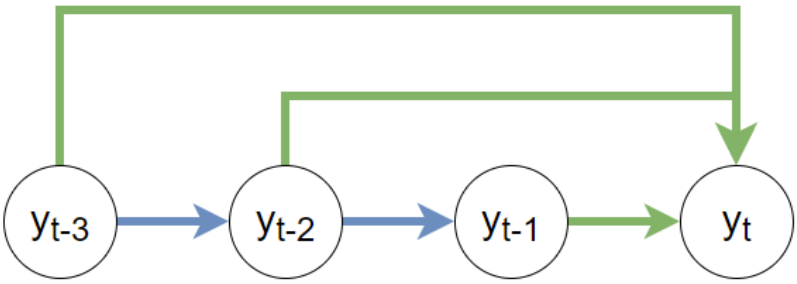

Abbildung 3-2 AR-Modell Funktionsweise.

Ein AR-Modell erster Ordnung bzw. *AR(1)* Modell wird durch die folgende Formel beschrieben [13 (Seite 23)][14 (Seite 61)]:

$$y_t = \mu + \phi_1 * y_{t-1} + e_t \qquad (3\text{-}1)$$

Für dieses Autoregressionsmodell mit dem Grad 1, entspricht der zu prognostizierende Parameterwert y_t der Summe aus dem Mittelwert der Zeitreihe μ, dem zum Zeitpunkt t-1 gewichteten Parameterwert $\Phi_1 * y_{t-1}$ und einem Fehlerwert e_t, welcher in der einschlägigen Literatur auch als „White-Noise“ bezeichnet wird [13] [14].

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 [13 (Seite 25-26)][14 (Seite 61)]:

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

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.

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 [13 (Seite 23)][14 (Seite 78)]. MA-Modelle prognostizieren nicht anhand der vergangenen Parameterwerte (so wie es AR-Modelle machen), sie prognostizieren anhand der vergangenen Fehlerwerte [13 (Seite 23)][14 (Seite 77-78)]. 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.

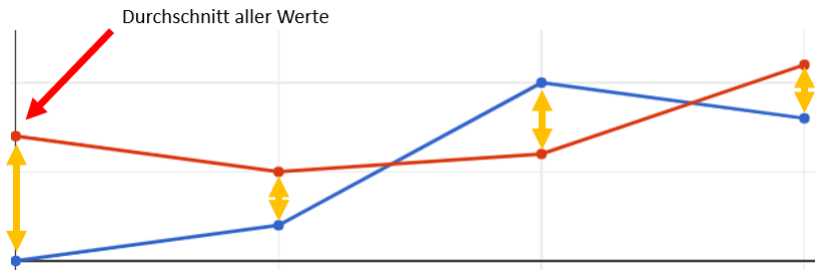

Abbildung 3-3 MA Bestimmung des initialen Prognosewerts und der folgenden Fehlerwerte.

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 [14 (Seite 78)]:

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

Dabei ist θ das Pendant zu Φ und somit ein lagabhängiger bzw. zeitabhängiger Gewichtungsfaktor.

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 [7][8][15-17]. 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:

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

Einsetzen der blau markierten Variable $\boldsymbol{y_{t-1}}$

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

Einsetzen der blau markierten Variable $\boldsymbol{y_{t-2}}$

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

...

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

Die hergeleitete Formel 3-4 entspricht einem *MA (∞)* Modell, also einer Summe gewichteter Fehlerwerte, was der Formel 3-3 entnommen werden kann. Der Parameter $\bar{\mu}$ beschreibt die Summe aller $\mu * \Phi$ Terme, welche für die Herleitung der Beziehung, zwischen AR-Modell und MA-Modell, nicht relevant sind. 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 [7][8][15]. Diese Kombination der AR und MA Terme kann der folgenden Formel 3-5 beispielhaft entnommen werden [7][8][15-17]:

$$y_t = \mu + \phi_1 * y_{t-1} + \cdots + \phi_p * y_{t-p} + \theta_1 * e_{t-1} + \cdots + \theta_q * e_{t-q} + e_t \tag{3-5}$$

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 [7][8][15-17]. 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 bzw. die Anzahl optionaler Integrationen der Zeitreihe und der Parameter *q* beschreibt den Grad des MA-Teils [7][8][15][16][17].
Somit wird ein *ARIMA(1, 0, 1)* Modell, mit einem AR-Teil, keinem I-Teil und einem MA-Teil, durch die folgende Formel 3-7 beschrieben:

$$y_t = \mu + \phi_1 * y_{t-1} + \theta_1 * e_{t-1} + e_t \tag{3-7}$$

Die thematisierte Komplexitätsreduktion führt dazu, dass nach der Validierung der Prognoseparameter eine schnelle Kalkulation der Prognose möglich ist [7][8]. 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) [7][8]. 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 [7].
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 [16][18-22]. 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 [18][20]. 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 [18-20]:

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

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 [18-20]. Der folgenden Formel 3-9 kann eine allgemeine Definition des SARIMA-Modells entnommen werden [18-20]:

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

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 Abschlussarbeit nicht nur statistische Methoden, sondern auch ein neuronales Netz zur Kurzzeitprognose der Turmschwingungskinematik verwendet werden, da dieses eventuell bessere Ergebnisse hervorbringt.
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 [9][11][12]. Eine detaillierte Dokumentation 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 [9][10][12]. 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 $s_p(t)$ dargestellt und *h(t)* gibt die unterschiedliche Gewichtung bzw. Relevanz verschiedener Zeiträume im Datensatz bzw. in der zu analysierenden Zeitreihe an [9][10][12]. Dadurch ist es möglich das initial für wirtschaftswissenschaftliche Fragestellungen entwickelte Netz (ähnlich wie beim ARIMA-Modell) auf diverse Fragestellungen zu adaptieren [11].
Im Rahmen dieser Abschlussarbeit wird Prophet gegenüber anderen neuronalen Netzwerken bevorzugt, da es hoch konfigurierbar ist und keine besonderen Anforderungen an den Datensatz in Bezug auf Stationarität stellt [9-12]. 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 [11].

4 Senvion Windkraftanlage

Die Senvion Windkraftanlage, mit einer Nennleistung von 3.37 MW, 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 Sensorbox zu entnehmen. Der konkrete Aufbau, Funktionsweise und Daten, welche von der Sensorbox, erfasst werden, werden in dem Kapitel 5 thematisiert.

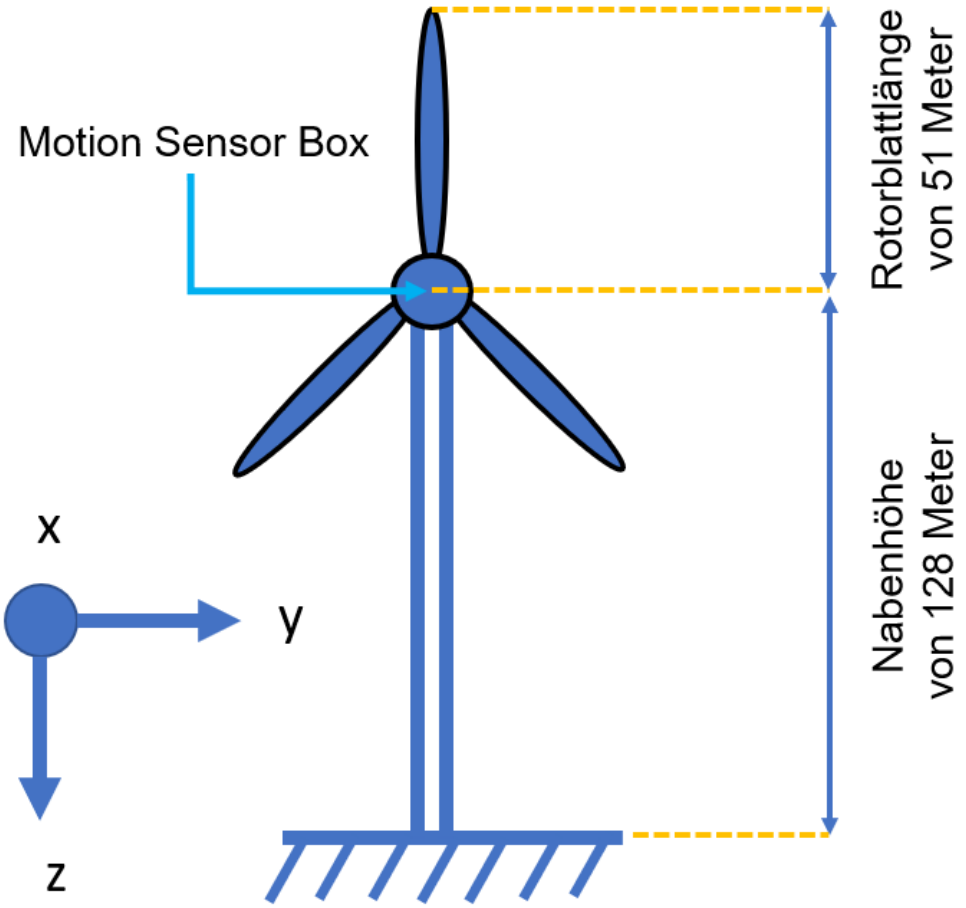

Abbildung 4-1 Skizze der Senvion Windkraftanlage.

5 Motion Sensor Box

Im Kapitel 4 wurde bereits beschrieben, dass an der Senvion-Windkraftanlage Motion Sensor Boxen angebracht sind, welche verschiedenste Parameter aufnehmen. Die Sensorboxen wurden von der Universität Bremen und Flucto GmbH gemeinsam entwickelt. Der folgenden Abbildung 5-1 ist eine solche Sensorbox zu entnehmen.

Abbildung 5-1 Motion Sensor Box.

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.
Dem folgenden Link kann ein Rohdatensatz entnommen werden, der grundlegend im Rahmen dieser Abschlussarbeit (in Kapitel 7) als Testdatensatz verwendet wurde:

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 Contact Elements for IoT Integration verwendet werden können.

6 Datenpipeline

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

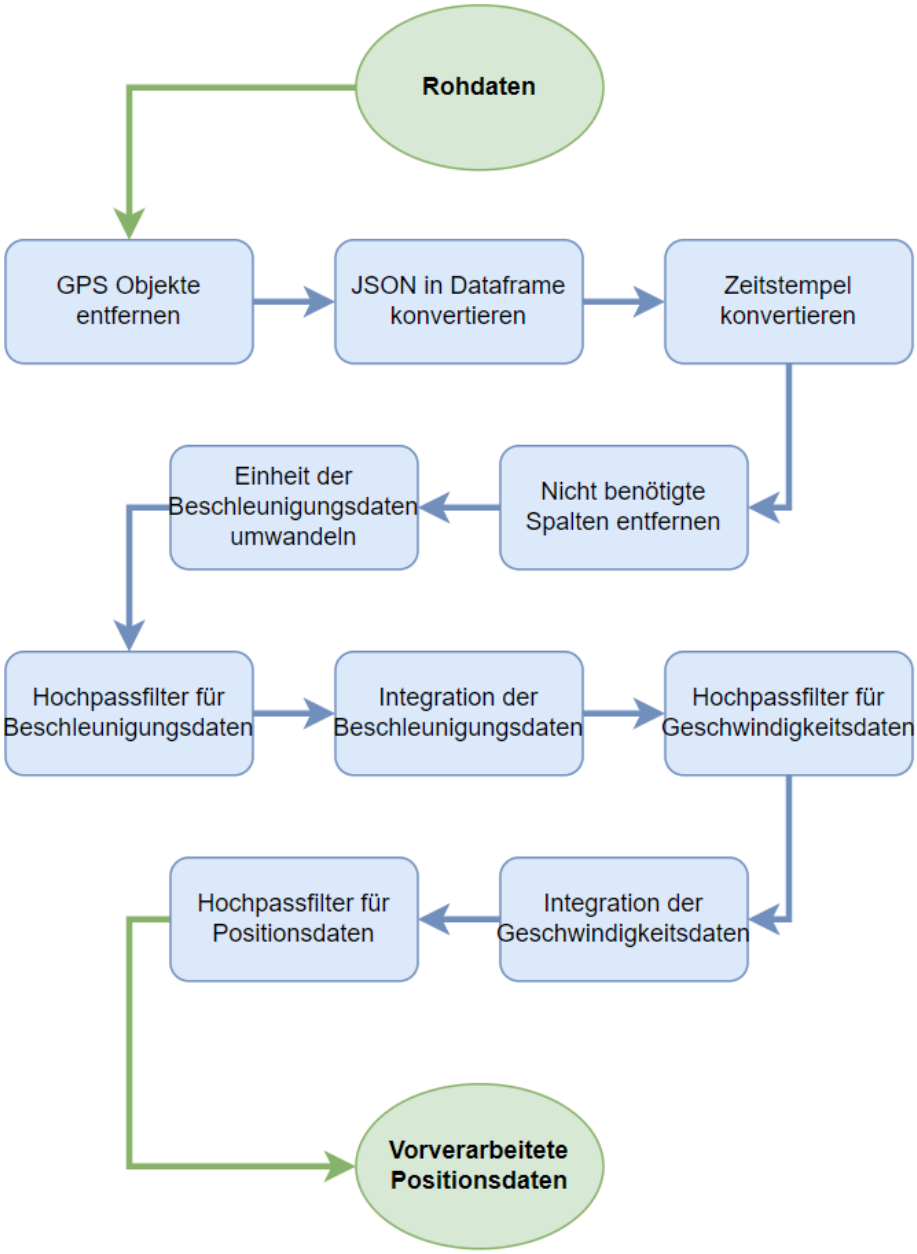

Abbildung 6-1 Datenpipeline.

GPS-Objekte entfernen

Im Kapitel 5 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-Objekte 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 des Datatime-Datenformats kann unter folgendem Link aufgerufen werden:

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

Nicht benötigte Spalten entfernen

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.

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

Da wie beschrieben im Rahmen dieser Forschungsarbeit die Prognose der Positionsdaten erforscht werden soll, kann der folgende physikalische Zusammenhang zwischen der Beschleunigung *a(t)*, Geschwindigkeit *v(t)* und Strecke *s(t)* 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 derselbe Butterworth Hochpassfilter angewandt (diesmal auf die Geschwindigkeitsdaten). 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

Es wird erneut derselbe Butterworth Hochpassfilter angewandt (diesmal auf die Positionsdaten). Die berechneten Positionsdaten werden dann als Eingabe der Prognosemodelle verwendet und können den folgenden Abbildungen 6-2, 6-3 und 6-4 beispielhaft entnommen werden.

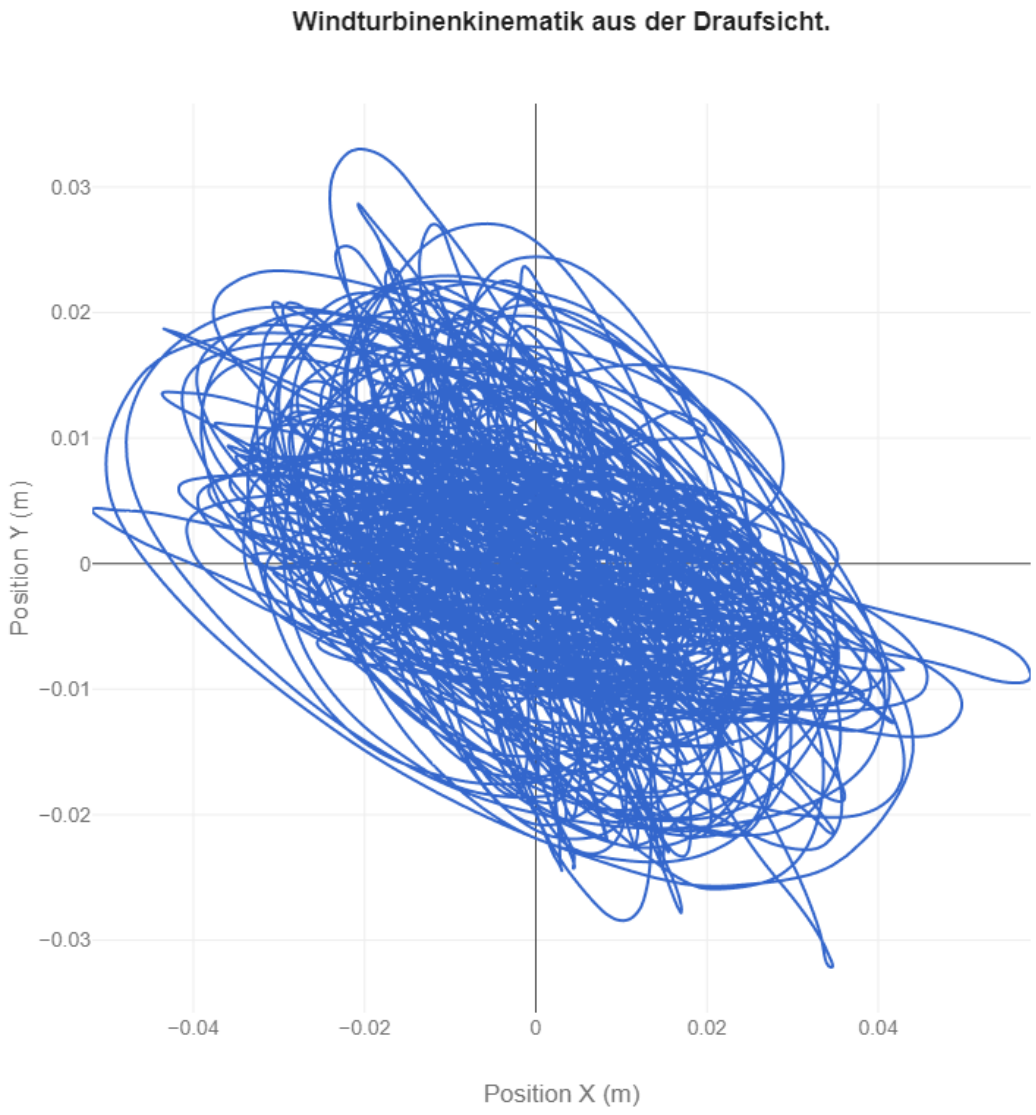

Abbildung 6-2 Windturbinenkinematik aus der Draufsicht.

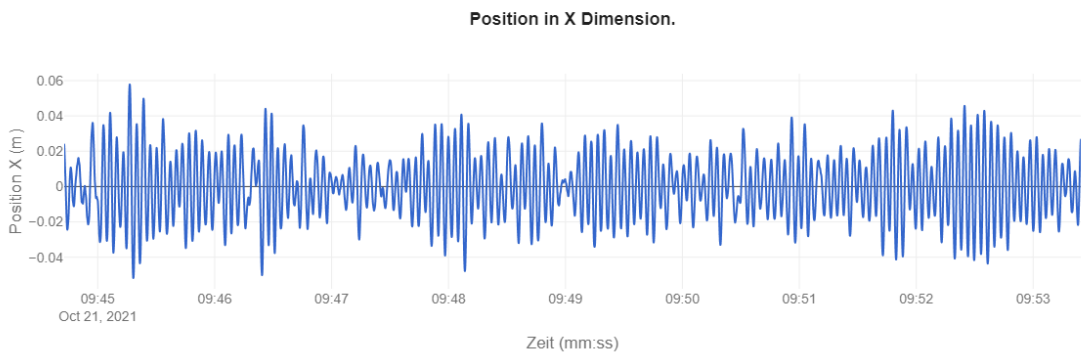

Abbildung 6-3 Positionsdaten in X Dimension (Vor-Zurück).

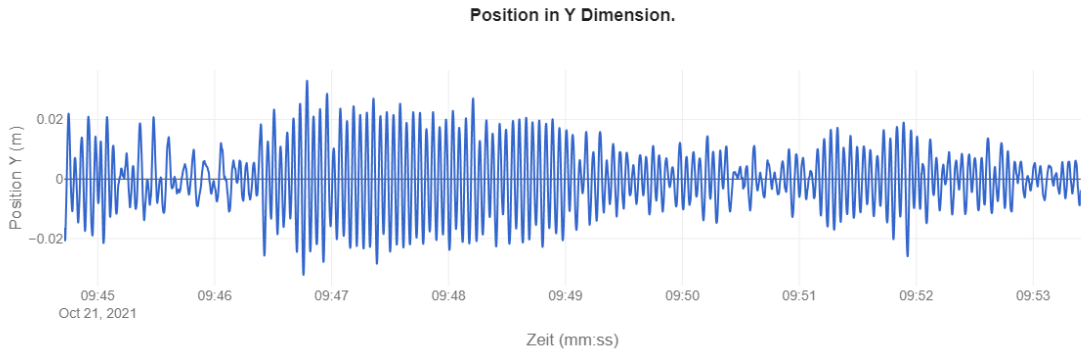

Abbildung 6-4 Positionsdaten in Y Dimension (Seite-Seite).

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 8.1 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. Dabei wird, wie im Kapitel 5 kurz beschrieben, ein historischer Datensatz vom 21 Oktober 2021 verwendet, welcher eine Stunde umfasst und in 50 Hz vorliegt. Zur Bewertung der Genauigkeit wird der RMSE Wert verwendet, welcher in der einschlägigen Literatur häufig zur Genauigkeitsbestimmung verwendet wird [20][21][22].

Die Berechnung kann der folgenden Formel 7-1 entnommen werden [20][21][22]:

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

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 zunächst auf eine Frequenz von 1Hz geresampelt, um die benötigte Rechenkapazitäten zu minimieren. 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]. Mögliche Informationsverluste, Vorteile und Nachteile dieses Verarbeitungsschritts werden im Kapitel 9.1 diskutiert. 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 (roter Graph) für einen Zeitraum von 80 Sekunden und der zugehörige Testdatensatz (blauer Graph) entnommen werden.

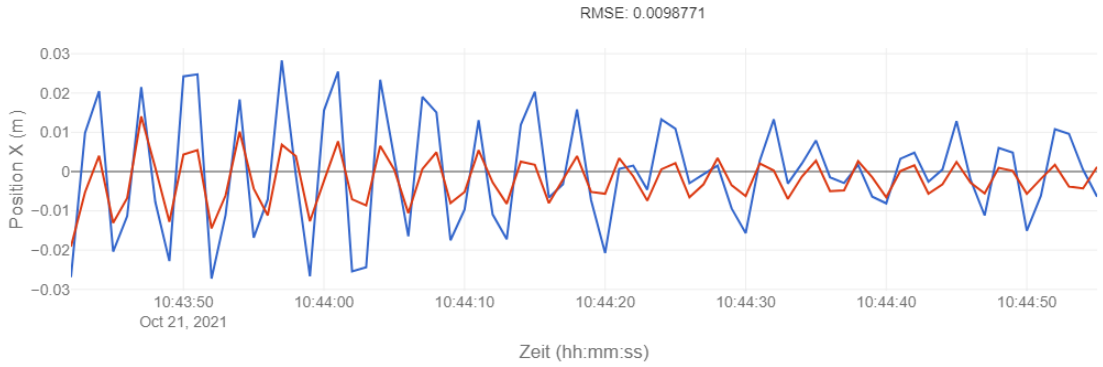

Abbildung 7-1 ARIMA(6,0,6) Prognose für Positionsdaten in X Dimension (Vor-Zurück).

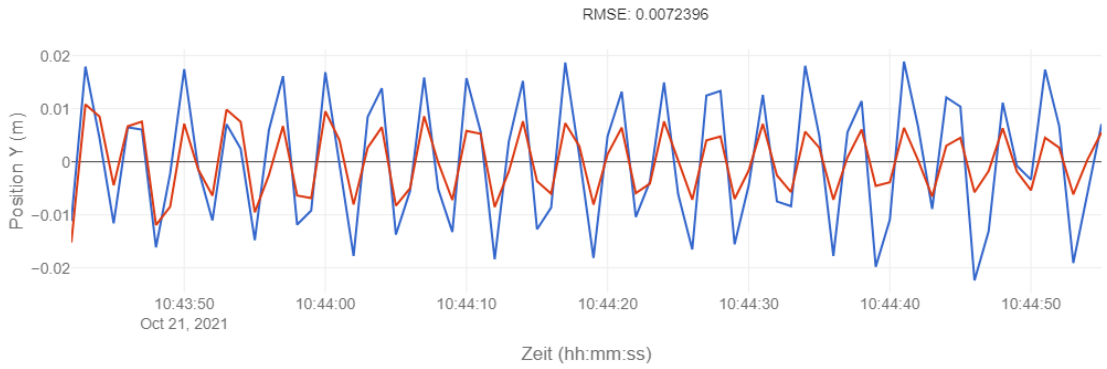

Abbildung 7-2 ARIMA(6,0,3) Prognose für Positionsdaten in Y Dimension (Seite-Seite).

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 zunächst auf eine Frequenz von 1Hz geresampelt, um die benötigte Rechenkapazitäten zu minimieren. 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]. Mögliche Informationsverluste, Vorteile und Nachteile dieses Verarbeitungsschritts werden im Kapitel 9.1 diskutiert. 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 (roter Graph) für einen Zeitraum von 80 Sekunden und der zugehörige Testdatensatz (blauer Graph) entnommen werden.

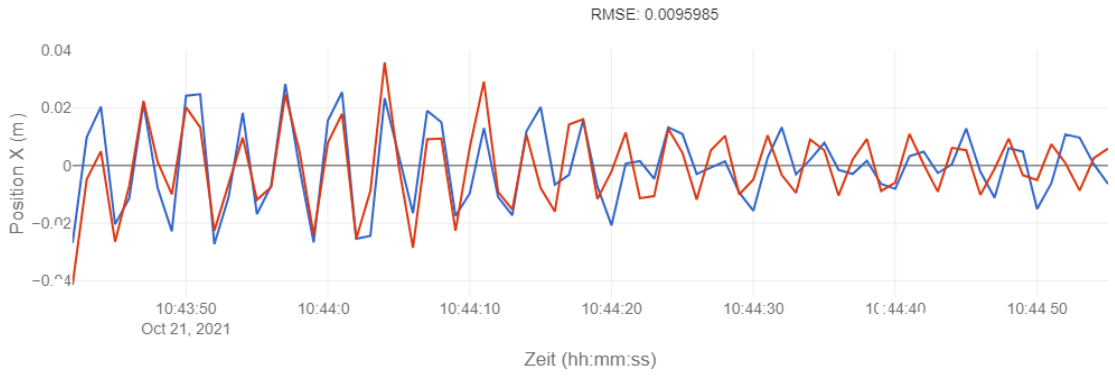

Abbildung 7-3 SARIMA(6,0,6)(6,0,6)$_7$ Prognose für Positionsdaten in X Dimension (Vor-Zurück).

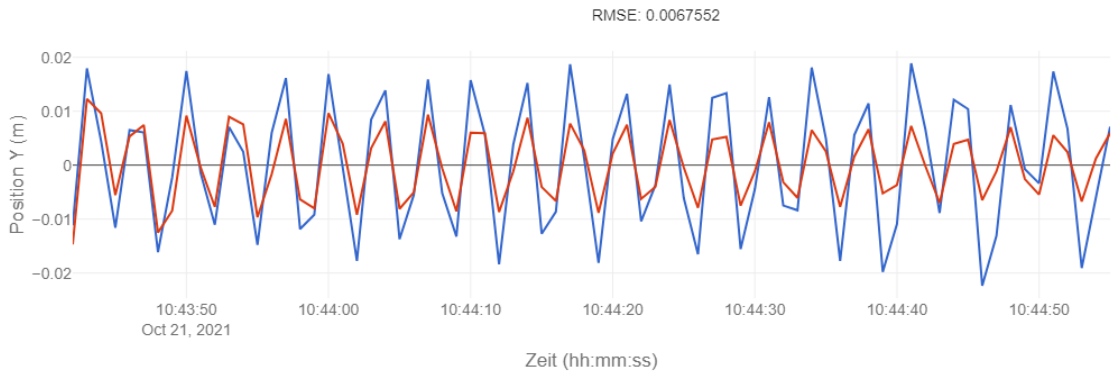

Abbildung 7-4 SARIMA(6,0,6)(6,0,6)$_7$ Prognose für Positionsdaten in Y Dimension (Seite-Seite).

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 zunächst auf eine Frequenz von 1Hz geresampelt, um die benötigte Rechenkapazitäten zu minimieren. 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]. Mögliche Informationsverluste, Vorteile und Nachteile dieses Verarbeitungsschritts werden im Kapitel 9.1 diskutiert. 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 (roter Graph) für einen Zeitraum von 80 Sekunden und der zugehörige Testdatensatz (blauer Graph) entnommen werden.

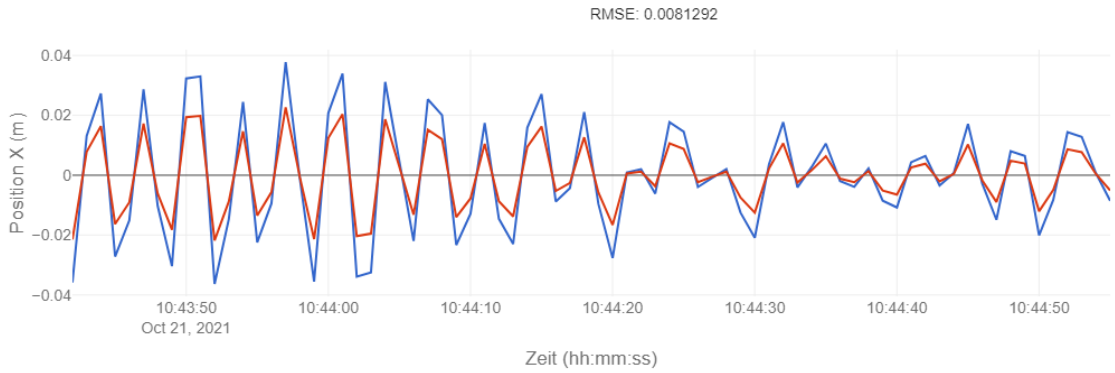

Abbildung 7-5 Prophet Prognose für Positionsdaten in X Dimension (Vor-Zurück).

Abbildung 7-6 Prophet Prognose für Positionsdaten in Y Dimension (Seite-Seite).

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 7-2 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 (Vor-Zurück)	Position in Y Dimension (Seite-Seite)
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:

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

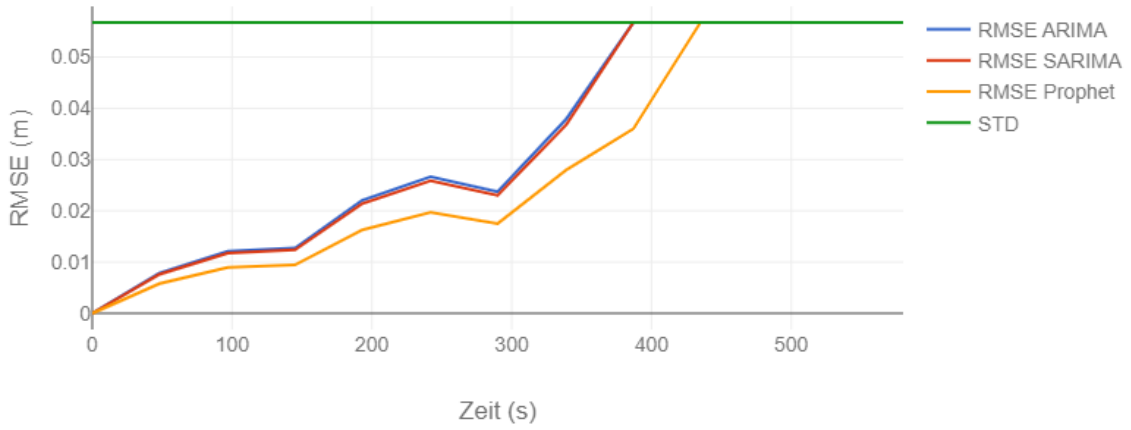

Abbildung 7-7 Prognosegenauigkeit der verschiedenen Modelle für Position in X Dimension (Vor-Zurück).

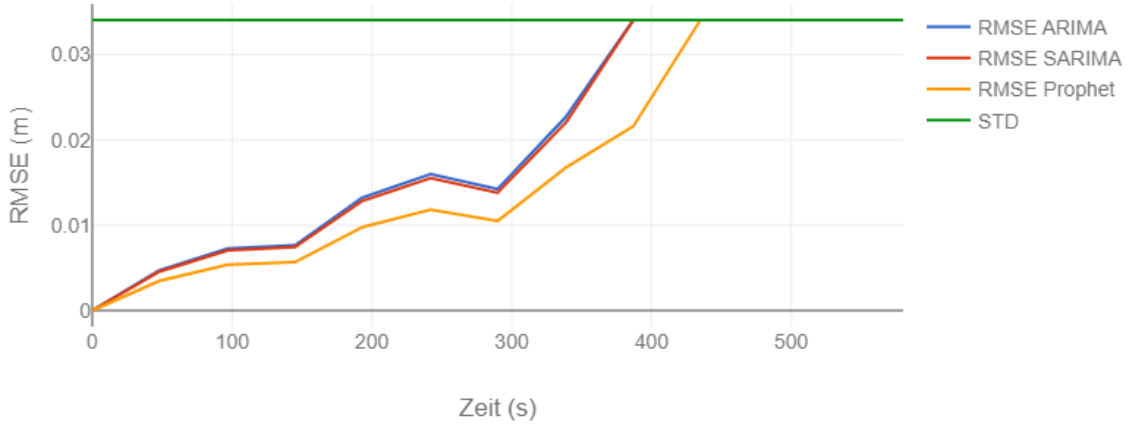

Abbildung 7-8 Prognosegenauigkeit der verschiedenen Modelle für Position in Y Dimension (Seite-Seite).

Die Abbildungen zeigen, dass alle drei Prognosemodelle für die Position in X Dimension und für die Position in Y Dimension eine gute Prognose für einen Zeitraum von bis zu ca. 150 Sekunden liefern. Dabei ist das SARIMA-Modell ähnlich genau in der Prognose wie das ARIMA-Modell und das Prophet-Prognosemodell ist signifikant genauer. Somit kann der Prognosemodellvergleich 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 (für die gegebene Fragestellung) 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 8.2 thematisiert.

7.5 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 drei Unterkapitel behandeln die theoretische Grundlage des Optimierungsansatzes, die praktische Umsetzung und die Auswertung der Ergebnisse.

7.5.1 Theoretische Grundlage

Der folgenden Abbildung 7-9 kann beispielhaft entnommen werden, dass die Positionsdaten aus der additiven Überlagerung einer Grundfunktion und einem White-Noise bestehen. Bei dem Training der Prognosemodelle durch diese insgesamt rauschende Funktion, kommt es zu einem Overfitting des Modells. Unter Overfitting versteht man unter anderem, dass das Prognosemodell im Trainingsvorgang zu stark auf das Rauschen trainiert wird und weniger auf die zugrundeliegende Funktion, welche rot markiert ist. Indem man das Rauschen entfernt und lediglich die zugrundeliegende Funktion zum Training des Prognosemodells verwendet wird, kann ein Overfitting vermieden werden und dies kann zu besseren Prognoseergebnissen (selbst für rauschende Testdaten) führen.

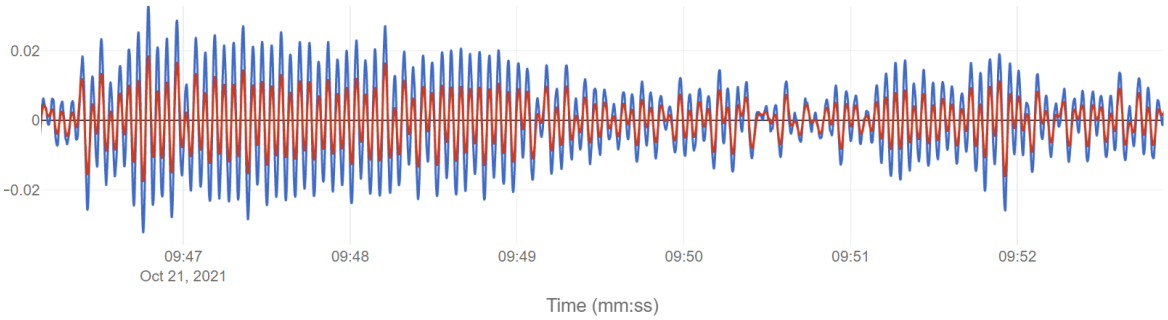

Abbildung 7-9 Beispielhafte additive Signalüberlagerung.

7.5.2 Umsetzung

Die im Kapitel 6 beschriebenen Positionsdaten werden als Eingabe des Prognosemodells verwendet. Dabei wird die Zeitreihe zunächst auf eine Frequenz von 1Hz geresampelt, um die benötigte Rechenkapazitäten zu minimieren. 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]. Mögliche Informationsverluste, Vorteile und Nachteile dieses Verarbeitungsschritts werden im Kapitel 9.1 diskutiert. Danach wird die Zeitreihe in einen Trainingsdatensatz und einen Testdatensatz aufgeteilt. Folgend wird, wie im Kapitel 7.5.1 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 7-3 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 7-10 entnommen werden.

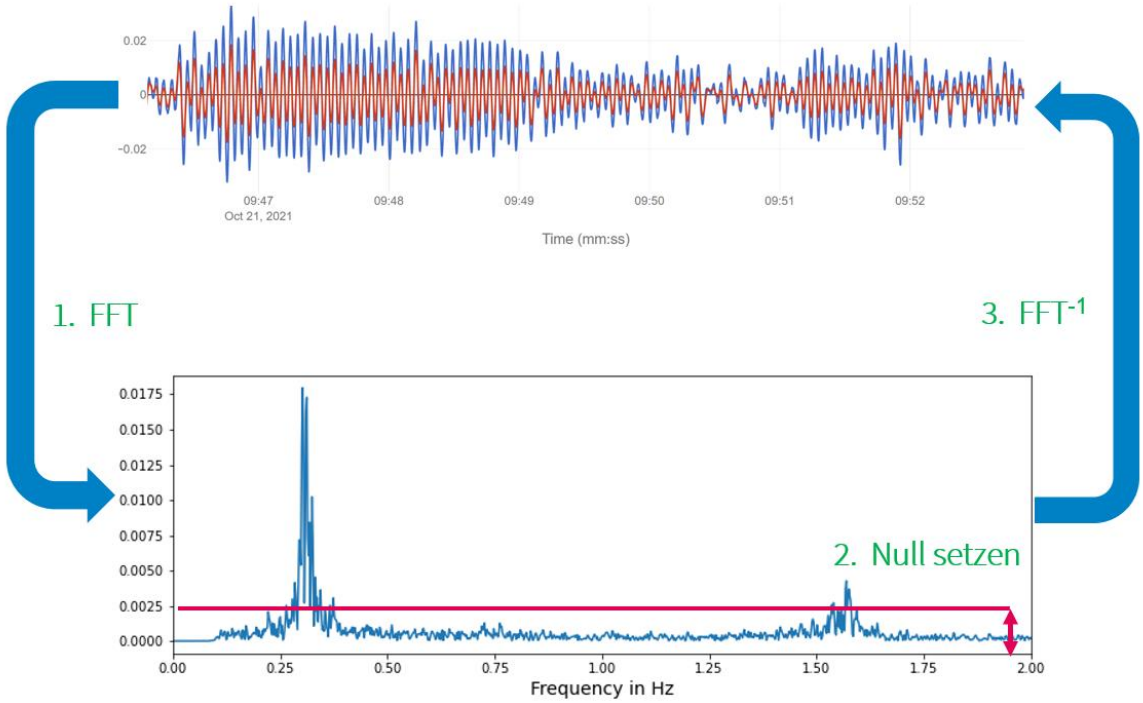

Abbildung 7-10 Beispielhafte Fourier Transformation zur Rauschentfernung.

Andere Studien haben Fourier Transformationen im Zusammenhang mit verwandten Datensätzen verwendet, um die Kinematik der Windenergieanlagen beschreiben 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 Maximum 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.

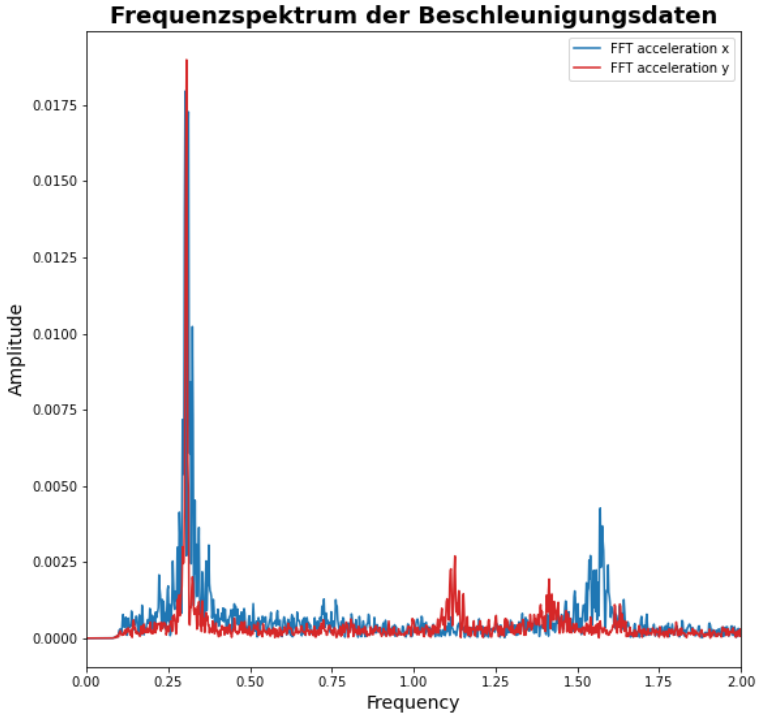

Abbildung 7-11 Frequenzspektrum der Beschleunigungsdaten der Senvion-Anlage.

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.

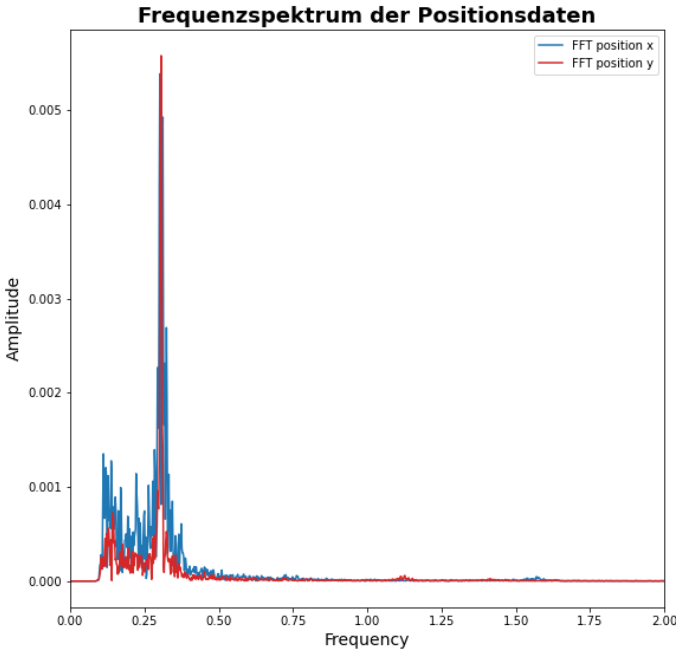

Abbildung 7-12 Frequenzspektrum der Positionsdaten der Senvion-Anlage.

Auch bei den Positionsdaten ist ein Maximum bei ca. 0.27 Hz zu erkennen. Der Abbildung 7-12 ist zu entnehmen, dass die Frequenzen mit hoher Amplitude unter dem Schwellwert von 0.6 Hz liegen. Somit bietet sich ein Tiefpassfilter an, welcher Frequenzen über der definierten Grenzfrequenz abdämpft bzw. vollständig entfernt. Den folgenden zwei Abbildungen 7-13 und 7-14, kann das Signal inklusive White-Noise Überlagerung (blau) und das gefilterte Signal (rot) entnommen werden.

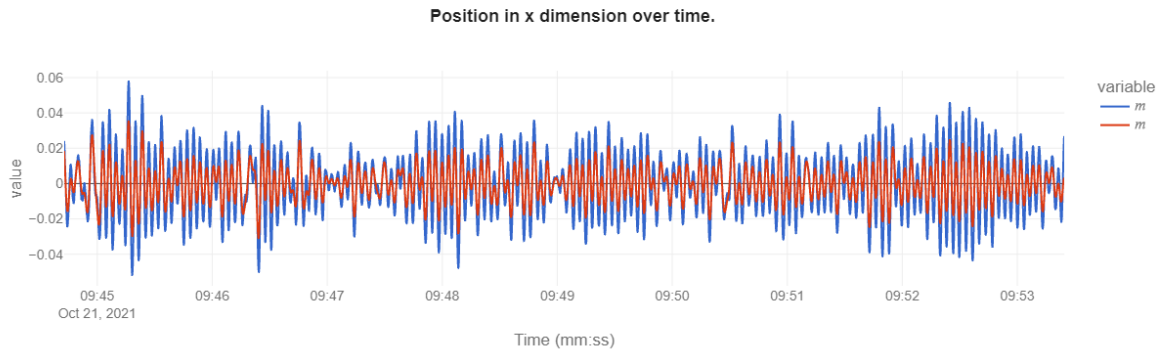

Abbildung 7-13 Frequenzspektrum der Beschleunigungsdaten der Senvion-Anlage.

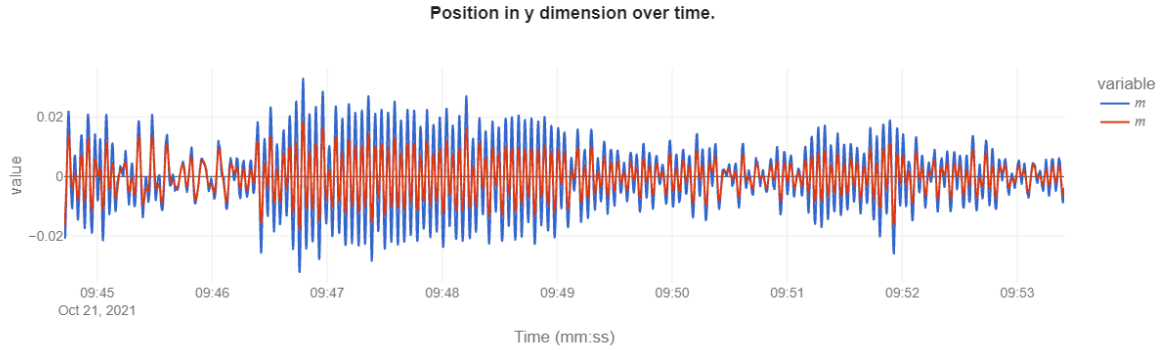

Abbildung 7-14 Frequenzspektrum der Beschleunigungsdaten der Senvion-Anlage.

7.5.3 Auswertung

Der folgenden Abbildung 7-15 und 7-16 ist das Ergebnis der verschiedenen Prognosemodelle zu entnehmen, wenn diese mit Tiefpassgefilterten Positionsdaten trainiert wurden. Bei einem Vergleich mit den Abbildungen 7-7 und 7-8, kann festgehalten werden, dass die Prognose für kurze Zeiträume geringfügig genauer ist, aber für lange Zeiträume statistisch signifikant genauere Ergebnisse liefern. Somit kann die Hypothese, die im Kapitel 7.5.1 formuliert wurde, bestätigt werden, dass das Training mit den gefilterten Positionsdaten, dazu führt, dass die Prognosemodelle weniger anfällig für ein Overfitting sind und somit bessere Prognoseergebnisse liefern können.

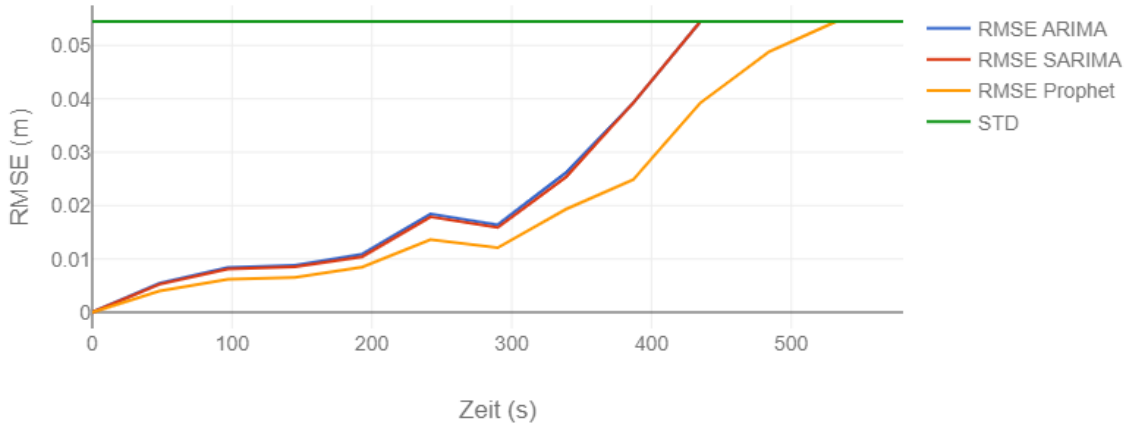

Abbildung 7-15 Frequenzspektrum der Beschleunigungsdaten der Senvion-Anlage.

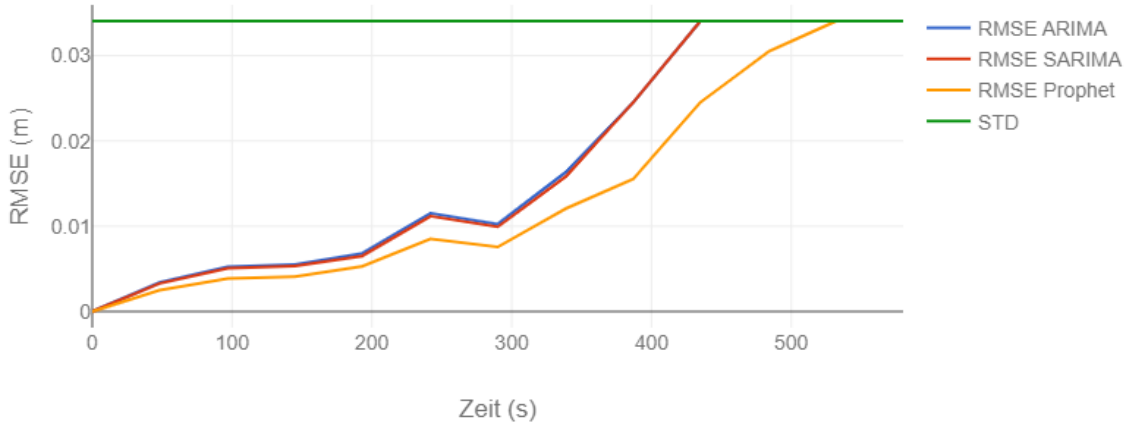

Abbildung 7-16 Frequenzspektrum der Beschleunigungsdaten der Senvion-Anlage.

8 Contact Elements Integration

Contact Elements for IoT ist die IoT Lösung von Contact Software GmbH. Die Abkürzung IoT steht für „Internet of Things“ und beschreibt die digitale Vernetzung von z.B. Maschinen und Sensoren 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.

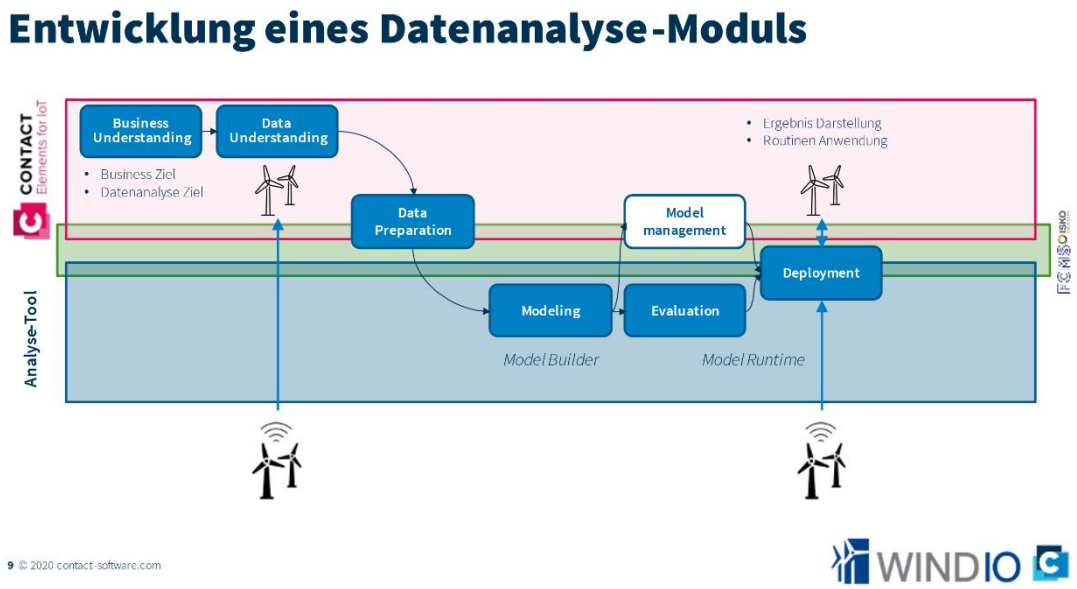

Abbildung 8-1 Contact Elements for IoT Entwicklung eines Datenanalyse-Moduls [CONTACT].

Die Integration der Prognosemodelle und die Implementierung der Datenpipeline, lassen sich in die thematischen Blöcke „Data Preparation“ und „Modeling“ einordnen. In den folgenden zwei Unterkapiteln werden die Implementierung der Datenpipeline und die Echtzeitprognose der Turmschwingungskinematik mithilfe des ARIMA-Modells und Prophet neuronalen Netzwerks thematisiert.

8.1 Implementierung der Datenpipeline

Zur Implementierung der Datenpipeline in Contact Elements for IoT, wird das Automatisierungstool verwendet. Dieses Werkzeug ermöglicht es bestehende Codebausteine zu verwenden oder sogar neue zu erstellen, aus denen dann komplette Automatisierungsprozesse zusammengestellt werden können. Da es zum aktuellen Zeitpunkt keine Livedatenübertragung von der Senvion-Windenergieanalage zum kommerziellen Digitalen Zwilling gibt, wird aus historischen Messdaten ein künstlicher Livedatenstrom generiert. Die Automatisierung zur Implementierung der Datenpipeline kann der folgenden Abbildung 8-2 entnommen werden.

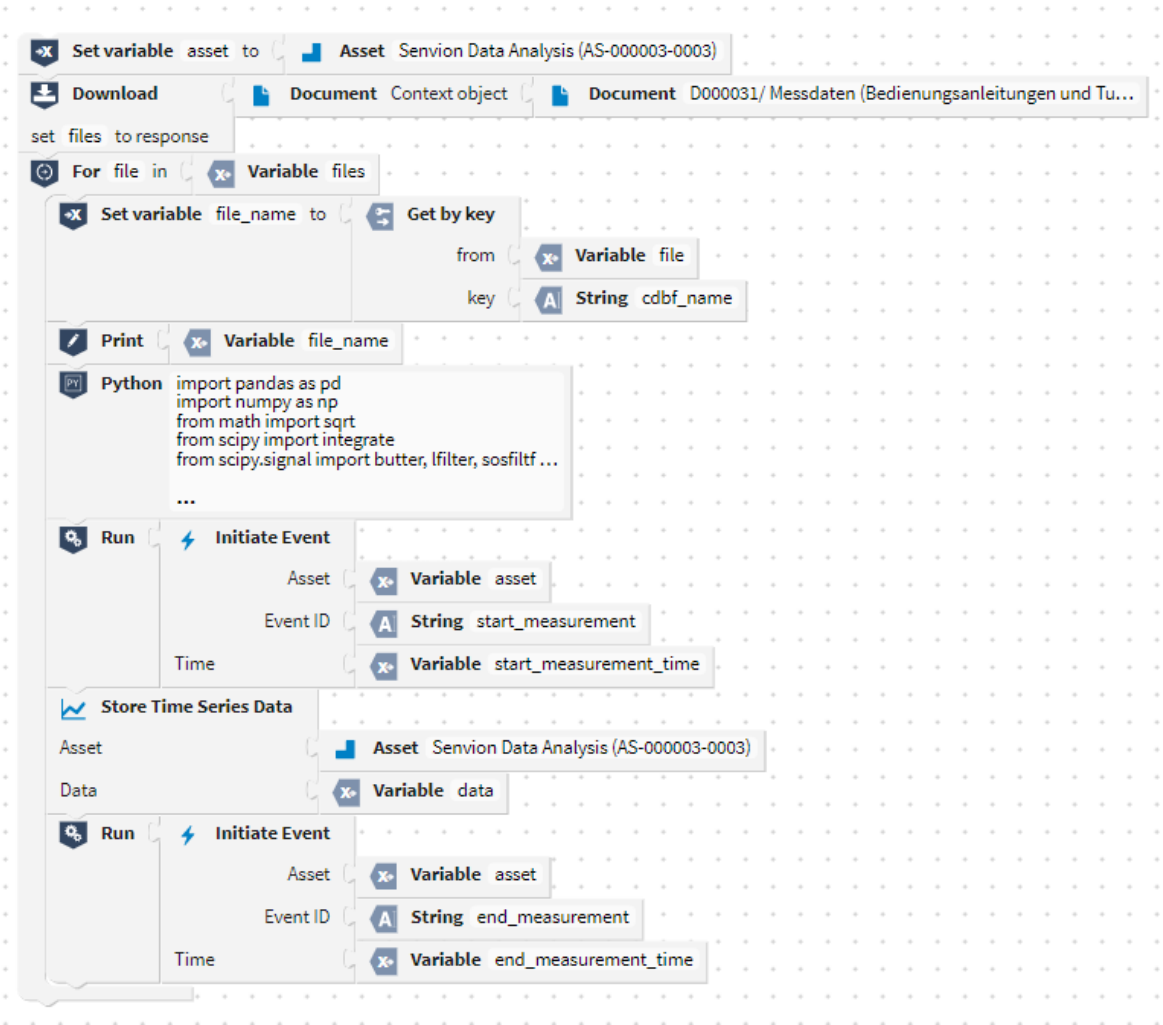

Abbildung 8-2 Automatisierung zur Implementierung der Datenpipeline.

Zunächst wird eine im System abgelegte CSV-Datei eingelesen, welche die historischen Messdaten enthält. Um die im Anschluss verarbeiteten Daten und die zugehörige Prognose in einem gemeinsamen Vorgang abzulegen, ist es notwendig zwei Ereignisse bzw. Events (start_measurement und end_measurement) zu definieren, welche diesen Vorgang starten und beenden. Vor diesen beiden Ereignissen befindet sich ein Pythonbaustein, der es einem ermöglicht eigenen Code einzufügen. Die Datenverarbeitung, wie sie im Kapitel 6 thematisiert wurde wird grundlegend durch diesen Baustein umgesetzt. Innerhalb der beiden Ereignisse werden die verarbeiteten Daten in einer Zeitreihe abgespeichert.

8.2 Echtzeitprognose der Turmschwingungskinematik

Auch zur Implementierung der Prognosemodelle wird das Automatisierungstool verwendet. Somit ist es möglich die erstellten Bausteine im Kontext anderer Fragestellungen wiederzuverwenden.

Es bestehen mehrere Möglichkeiten diese Implementierung praktisch umzusetzen. Im Rahmen dieser Integration wird ein von der Datenpipeline unabhängiges Automatisierungsskript erstellt, womit diese Prognosemodelle auch unabhängig von der Datenpipeline verwendet bzw. wiederverwendet werden können. Der folgenden Abbildung 8-3 kann das Prognosemodell-Automatisierungsskript entnommen werden.

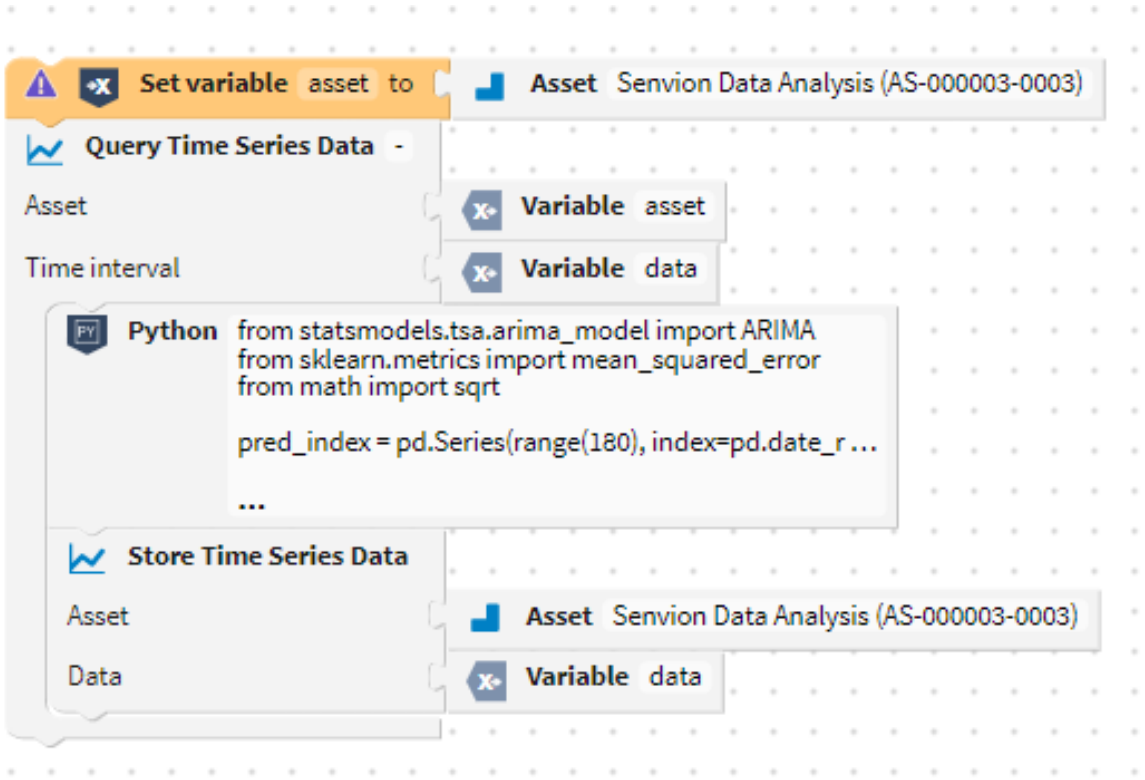

Abbildung 8-3 Automatisierung zur Implementierung der Prognosemodelle.

Zunächst werden die durch die Datenpipeline-Automatisierung verarbeiteten Positionsdaten eingelesen und dem eingeschlossenen Pythonbaustein als Eingabe zur Verfügung gestellt. Der Pythonbaustein enthält den notwendigen Code, um die Prognose zu berechnen. Prinzipiell kann hier der Code aller drei vorgestellten Prognosemodelle (Links sind dem Kapitel 7 zu entnehmen) eingefügt werden.

9 Diskussion

In den folgenden zwei Unterkapiteln werden die Prognoseergebnisse bewertet bzw. interpretiert und mögliche Probleme des Resampelns diskutiert.

9.1 Prognoseergebnisse

Im Kapitel 7.4 wurden die Prognosemodellergebnisse (aller drei Modelle) für unterschiedliche Prognosezeiträume ausgewertet. Als Metrik wurde der RMSE verwendet, um die Genauigkeit der Prognoseergebnisse zu bewerten. Den Abbildungen 7-7 und 7-8 konnte entnommen werden, dass das ARIMA-Modell und SARIMA-Modell fast identisch in der Genauigkeit der Prognose sind und das Prophet neuronale Netz signifikant genauere Ergebnisse kalkuliert.
Folgend wurde im Kapitel 7.5 die mögliche Problematik des Overfittings thematisiert und wie mithilfe eines Tiefpassfilters das Rauschen der Positionsdaten entfernt bzw. rausgefiltert werden kann. Den Abbildungen 7-15 und 7-16 konnte entnommen werden, dass eine Tiefpassfilterung der Positionsdaten zu einer genaueren Prognose, durch die drei Modelle, besonders für lange Zeiträume (ab 150 Sekunden) führt. Wie beschrieben, wurde in diesem Zusammenhang eine Grenzfrequenz von 0.6 Hz verwendet. Dieser Schwellwert für den Tiefpassfilter wurde auf Grundlage der Abbildung 7-12 gewählt, da ab einer Frequenz von 0.6 Hz nur noch geringe Amplitudenwerte festzustellen sind. Dieser Schwellwert könnte jedoch auch anders gewählt werden und die daraus resultierenden Einflüsse auf die Prognosegenauigkeit sollten im Rahmen weiterer wissenschaftlicher Arbeiten erforscht werden.
Die Turmschwingungskinematik der Windenergieanlage wird maßgebend durch äußere Einflüsse, wie z.B. die Windgeschwindigkeit, beeinflusst. Jedoch wurden diese, im Rahmen dieser Abschlussarbeit, als zufällige Störgrößen bzw. stochastische Prozesse aufgefasst, da nicht die entsprechenden Daten vorliegen. Weitere Forschungsarbeiten könnten sich mit der Korrelation zwischen diesen Umwelteinflüssen und der Turmschwingungskinematik beschäftigen, sodass dann auf Grundlage dieser Erkenntnisse ein zweiter Optimierungsansatz implementiert werden kann.

9.2 Resampeln der Positionsdaten

Im Kontext der Prognoseberechnung in den Kapiteln 7.1.1, 7.2.1, 7.3.1 und 7.5.1 wurden die Positionsdaten, welche in 50 Hz vorliegen, auf eine Frequenz von 1 Hz geresampelt. Dieser zusätzliche Verarbeitungsschritt soll die benötigte Rechenkapazität minimieren, um eine schnelle Berechnung der Prognose zu gewährleisten. Außerdem wurde erläutert, dass nicht die Vorhersage jeder einzelnen Bewegung bei einer Frequenz von 50 Hz von Interesse ist, sondern die Anhäufung bzw. Ansammlung von kinematischen Ereignissen [23]. Durch diesen Verarbeitungsschritt kann es jedoch zu einem Informationsverlust kommen, da Maxima und Minima geglättet werden. Der folgenden Abbildung 9.1 sind die Positionsdaten in X Dimension (Vor-Zurück) in 50 Hz und 1 Hz zu entnehmen.

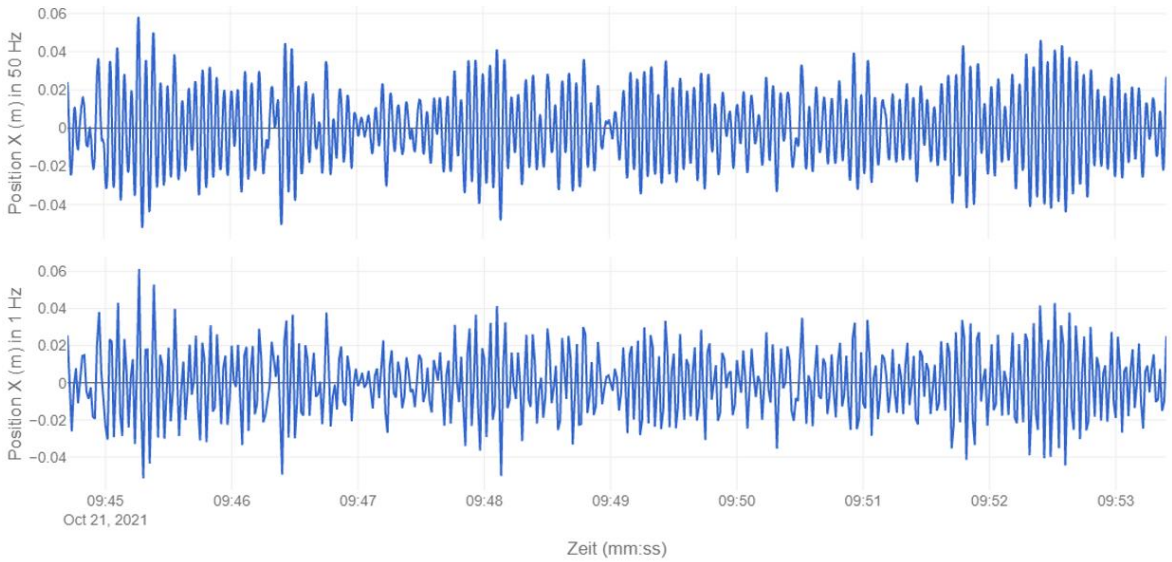

Abbildung 8-3 Automatisierung zur Implementierung der Prognosemodelle.

Der Abbildung ist zunächst rein grafisch zu entnehmen, dass es keinen signifikanten Unterschied zwischen diesen beiden Graphen gibt. Außerdem beträgt der RMSE Wert zwischen diesen beiden 0.002 m.

Die grafische Veranschaulichung und der RMSE Wert bestätigen die Behauptung, dass es für die gegebene Fragestellung und den analysierten Datensatz keinen signifikanten Unterschied zwischen den beiden Auflösungen (1 Hz und 50 Hz) gibt. Ob diese Behauptung auch auf andere Datensätze zutrifft, kann im Rahmen dieser Abschlussarbeit nicht abschließend beantwortet werden und sollte in anderen wissenschaftlichen Arbeiten näher erforscht werden.

10 Fazit

Das übergeordnete Ziel dieser Abschlussarbeit ist es die wissenschaftliche Fragestellung zu beantworten, wie genau die Turmschwingungskinematik, von Onshore-Windenergieanlagen, für die nächsten Sekunden und Minuten vorhergesagt werden kann.

Dazu wurden im Kapitel 7 drei Prognosemodelle (ARIMA, SARIMA und Prophet) implementiert und die Genauigkeit der Prognose für verschiedene Zeiträume, mithilfe der RMSE Metrik, ausgewertet. Den Abbildungen 7-7 und 7-8 konnte entnommen werden, dass das Prophet Modell die besten Ergebnisse hervorbringt und das ARIMA und SARIMA Modell fast identisch in der Prognosegenauigkeit sind. Da das SARIMA Modell, aufgrund der zusätzlichen saisonalen Parameter, deutlich mehr Rechenkapazität benötigt, kann festgehalten werden, dass das SARIMA Modell im Kontext der gegebenen Fragestellung ungeeignet ist.

Im Kapitel 7.5 wurde die Problematik des Overfittings thematisiert und ein Optimierungsansatz implementiert. Durch die Tiefpassfilterung der Trainingsdaten, mit einer Grenzfrequenz von 0.6 Hz, ist es möglich die Prognoseergebnisse besonders für lange Zeiträume (ab 150 Sekunde) zu verbessern, da das Modell auf die zugrundeliegende Funktion der Positionsdaten trainiert wird und das Rauschen der Daten minimiert wird. Die Wahl der Grenzfrequenz wurde im Kapitel 9.1 diskutiert und aufgezeigt, dass die Wahl dieses Schwellwerts und der Einfluss auf die Prognosemodellgenauigkeit weitergehend erforscht werden sollte. Diese Ergebnisse basieren auf der Annahme, dass es sich bei den Umwelteinflüssen, wie z.B. der Windgeschwindigkeit um stochastische Prozesse handelt, da in diesem Kontext keine Daten vorliegen.

Das Kapitel 8 thematisiert dann die Implementierung der Datenpipeline zur Vorverarbeitung eines kontinuierlichen Rohdatenstroms und die Implementierung der Prognosemodelle in Contact Elements for IoT. Durch das Verwenden des Automatisierungstools ist es möglich diese Komponenten auf unterschiedliche Fragestellungen zu adaptieren und wiederzuverwenden. Außerdem stehen die im Rahmen dieser Abschlussarbeit entwickelten Skripte auf GitHub unter einer MIT Lizenz frei zur Verfügung.

https://github.com/WindIO-Bachelorthesis/Shortterm_Forecast

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] 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.

[8] 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.

[9] 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.

[10] 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.

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

[12] 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.

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

[14] 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

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[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] 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.

[22] 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.

[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.