Enhancing Heliostat Calibration in Solar Tower Power Plants: A Novel Dataset Evaluation Metric and Hybrid Kinematic Modelling Technique

Verbesserung der Heliostat-Kalibrierung in Solarturmkraftwerken: Eine neue Metrik zur Datensatzanalyse und Technik zur Hybriden Kinematischen Modellierung

Der Fakultät für Maschinenwesen der Rheinisch-Westfälischen Technischen Hochschule Aachen vorgelegte Abschlussarbeit zur Erlangung des akademischen Grades "Master Of Science"

von
Moritz Leibauer

Prüfer: Univ.-Prof. Dr.-Ing. Robert Pitz-Paal

Betreuer: Dr. Ing. Max Pargmann
Dipl. Ing. Vincent Nettelroth

Abgabedatum: May 25, 2023

"There are a thousand no's for every yes."

Abstrakt

Die Energieausbeute von Heliostatfeldern wird hauptsächlich durch die Zielgenauigkeit der Heliostaten beeinflusst. In dieser Arbeit wird ein neuartiger Ansatz zur Optimierung der Zielgenauigkeit vorgestellt, der dynamisches Heliostat-Verhalten berücksichtigt und die Anzahl benötigter Trainingsdaten reduziert.
Der Industrie-Standard der Verwendung eines kinematischen Modells mit steifen Körpern wird um eine Geometrieparameter-Optimierung durch ein Neuronales Netz erweitert. Mit der vorgestellten Methodik können dynamische Abweichungen vom idealen Verhalten aufgrund mehrerer Stellgrößen berücksichtigt werden. Durch die Reduktion der Problem-Komplexität vom gesamten kinematischen Verhalten auf die Vorhersage der Abweichungen vom idealen Verhalten, können einfache Netzwerk-Architekturen verwendet werden. Im Vergleich zur bisherigen Anwendung Neuronaler Netze entfällt die Notwendigkeit des Pretrainings zum Erlernen der Kinematik, und folglich wird die Anzahl benötigter Trainings-Datenpunkte reduziert. Die Kombination der Verwendung eines vortrainierten kinematischen Modells mit dynamischer Geometrieparameter-Anpassung, übertrifft die Zielgenauigkeit des derzeitigen Industrie-Standard für mehr als 95% der getesteten Daten.
Durch die Einführung einer Distanz-Metrik zwischen den Sonnenpositionen einzelner Kalibrierpunkte, wird eine Methodik zur Beurteilung der Abdeckung des Heliostatverhaltens durch gegebene Datensätze geschaffen. Diesem Prinzip folgend, wird ein Algorithmus zur Aufteilung von Datensätzen in Trainings-, Validierungs- und Testdaten eingeführt, der heuristisch die Abdeckung des Heliostatverhaltens, sowie die Datenpunktverteilung optimiert. Anhand von drei Datensätzen, die während des Betriebs am Solarturm des DLR in Jülich im Zeitraum zwischen Mai 2021 und Oktober 2022 gesammelt wurden, wird die Verwendung der mittels der Metrik erstellten Datenpunktverteilung evaluiert. Für zwei von drei Datensätzen werden mittlere Zielgenauigkeiten von unter 1mrad bei kleinen Trainingsdatensatzgrößen von 20 Trainings- und 20 Validierungsdaten erzielt.
Zukünftig können die gesammelten Erkenntnisse dazu genutzt werden, die Datenverteilung von Heliostatdatensätzen besser zu beurteilen.

Abstract

The efficiency of heliostat fields is mostly affected by the heliostat's alignment accuracies. A novel approach for optimizing these accuracies is introduced within this thesis, that includes dynamic heliostat behavior and reduces the amount of required training data.
The current state of the art approach of using a rigid-body kinematic model is extended by exchanging the model's geometry parameters by a neural network. Using this method allows the model to account for various dynamic impacts, such as structural bending. By excluding most of the ideal kinematic behavior from the neural network, the problem-complexity is reduced to computing deviations from the ideal behavior and thus applying simpler neural network architectures. In constrast to current neural network approaches, no pretraining on modelled data must thus be applied, which reduces the required amount of training data. The combination of using a pre-optimized kinematic model with dynamic geometry parameter adaptation by a neural network exceeds the current industry standards accuracies in more than 95% of the given test cases.
By introducing a distance-metric between calibration data points' solar positions, a new method for evaluating a dataset's coverage of the entire heliostat behavior is achieved. Following this principle, an algorithm for splitting datasets into training-, validation- and test-data is introduced, that heuristically optimizes the data point distribution and heliostat behavior coverage. By means of three datasets that were collected at the German Aerospace Center (DLR)'s solar tower in Jülich between May 2021 and October 2022, the introduced metrics benefits to model training are verified. For two out of three datasets the trained model achieves average heliostat alignment accuracies below 1mrad for the entire year starting from as little as 20 training- and 20 validation data points.
The obtained scientific insights in future can be applied to improve the evaluation of heliostat alignment dataset distribution as well as the performance of different heliostat models.

Contents

1. Introduction

The global energy sector is still dominated by fossil fuels, with alternative power sources only supplying about 20% [1] of the global energy demands. One promising approach for diversifying the energy production with renewable sources lies in the application of Solar Towers (STs), whose concentrator efficiency optimization is the focus of this thesis.

Recent events emphasized a possible lack of robustness in the energy supply of regions, where single energy sources hold high shares. In 2022 technical issues and maintenance forced a shutdown of 57% [2] of France's nuclear power generation capacity (equalling about 40% of France's total power generation capacity and 6% of Europe's power generation capacity). Meanwhile, a severe drought forced power cuts at the world's largest hydropower plant in China [3]. At the same time, the German shift from coal to natural gas induced soaring energy prices, when Russian gas exports stopped in the wake of the Ukrainian war, reducing the nation‘s economic growth by 1,8% [4]. Furthermore, the shift from fossil fuels to alternative sources, that is necessary to stop human induced climate change, will lead to a higher consumption in minerals and has started a global race to ensure mineral supply chains [5]. To stabilize the global energy supply the means of energy production thus will have to be diversified.

ST is mostly independent from rare earth elements [6] as well as being capable of storing energy directly as high temperature heat, reducing the required amounts of batteries for its operation. Therefore, a ST plant's material requirements differ greatly from competitors such as photovoltaik, making it a viable alternative for energy diversification.
Paramount to each ST plant is its collector‘s concentration accuracy. Collector alignment errors cause potentially dangerous structural burnups and lead to efficiency losses reducing plant efficiencies down to 58%-72% [7]. To improve the concentration accuracy, a digital twin of the plant's collector is created and calibrated on real world performance data. A virtual twin thereby represents a model, whose parameters are fitted to performance data from the real world object. Collector calibration spans the process of collector beam characterization (data gathering) with systems called Beam Characterization System (BCS), up to the gathered data evaluation and collector model training.

1.1. Motivation

A multitude of BCS [8] as well as collector models and model training techniques have been proposed, but still leave room for improvements. Every collector model's capability to predict events outside the training dataset depends on the model's abstraction capability. The major lever in model training therefore is the amount of collected data as well as the data's noise ratio. *Sarr et al. (2023)* [9] noticed a dependency between training data distribution and alignment model performance, but didn't present any further investigation. Therefore, no means to evaluate a data point's added value to a dataset have been published until today. The lack of sufficient referencing of data distributions poses a significant challenge in comparing the performance of different alignment model approaches.

Another impact on the model's abstraction capability is the model's completeness. The conventional training approach, which relies on numerical optimization for model parameter estimation, succeeds in learning from small datasets. However, it demonstrates limitations in being restricted to rigid-body mechanics and therefore is unable to include environmental influences and alignment-dependent factors such as structural bending induced by load. These neglected impacts, however, were found [10] to potentially cause major deviations on the concentrator's behavior. Neural networks can fix this shortcoming, but require larger datasets [11]. However, data acquisition to this date is time intensive and thus expensive [8]. Another issue when applying neural networks to ST systems is their power plant nature and therefore the high demand for system safety and predictability. Handling full concentrator control to an Artificial Intelligence (AI) might become a hurdle for obtaining permits from local regulators.

1.2. Aim

This thesis aims to answer two scientific questions. First, whether heliostat alignment data point values can be quantified and second, if the standard approach of kinematic heliostat modelling can be combined with modern Neural Network techniques to achieve dynamic behavior modelling on low amounts of data.

To improve the evaluation quality of alignment models, a stochastic approach to alignment training data distributions is pursued. A distance function between data points is introduced. Based on this distance function, dataset distributions can be stochastically quantified and thus compared. The found principles are furthermore applied to introduce a conservative estimation to an alignment model's performance on a given dataset and thus give more realistic expectations to its prediction capabilities. The same approach is additionally employed to demonstrate that a significantly smaller amount of model training data is necessary, when the training data is equally distributed based on distance.

A two-layered heliostat alignment model, that can account for dynamic behavior such as structural bending, wear or environmental impacts, is implemented into a fully differentiable environment [12]. This is achieved by using the standard kinematic model approach as first layer while enhancing its constant kinematic parameters in the second layer by using a dynamic disturbance model, that predicts deviations to the parameters based on given inputs such as the alignments orientation. The introduced performance estimation techniques are eventually applied to compare the suggested alignment model to the standard kinematic model approach.

1.3. Structure

In section 1 the research topic's motivation, aim and structure are introduced, while section 2 summarizes the relevant scientific foundation and focuses on constructing a terminology framework on which this work is based upon.

Section 3 gives an evaluation of the results and approaches of related research on the topic of heliostat alignment optimization. Special focus is given on dataset distributions and result presentation.

The following chapters section 4, section 5 on introducing the applied methods. First, heliostat alignment datasets are stochastically analyzed and a metric for evaluating heliostat alignment models is derived in section 4. Thereafter a new approach for heliostat alignment models is introduced in section 5.

The investigated datasets are introduced in section 6. Section 7 introduces the pursued experiments and their results. The findings are then discussed in section 8.

Eventually, section 9 summarizes the thesis' results and section 10 provides an outlook into further research opportunities.

2. Scientific Foundation

This research is set within the overlapping scientific areas of concentrated solar power plants, digital twin modelling, machine learning and artificial intelligence. Therefore, these topics' theoretical foundation, relevant to this thesis, is summarized within the next sections. Furthermore, a common terminology is pursued, as the above-mentioned research fields follow different approaches, which results in their terminology being ambiguous.

2.1. CSP Power Plants

Concentrated Solar Power (CSP) systems can be divided into "Trough" and "Tower" systems. A ST plant collects solar radiation and concentrates it onto a receiver area. Consequentially, the receiver energy density, known as flux, exceeds its surroundings. Once concentrated, the energy can be transferred to a reactor as heat and either be directly converted into electricity, stored in a heat accumulator or used as process heat (e.g. to create solar fuels).

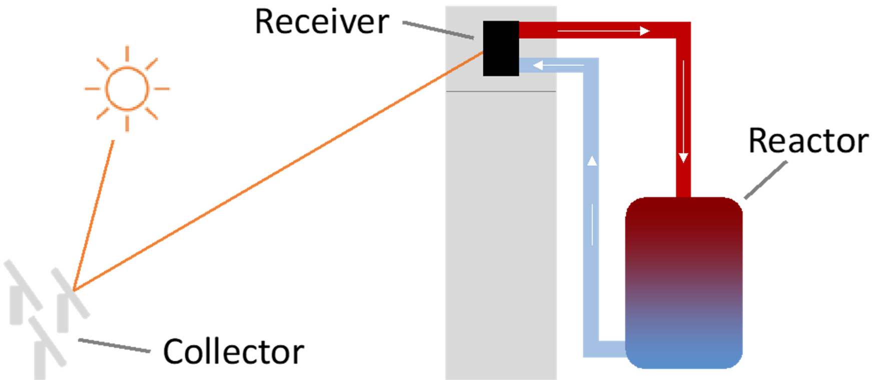

Figure 1: Principle of a Solar Tower system. The solar radiation is redirected by a collector onto a central receiver. The resulting high energy density is then applied to industrial processes (e.g. power generation) within a reactor.

A major component of each ST plant is its collector. Relevant in the context of this thesis are CSP Tower systems, shown in Figure 1 and thus heliostat field collectors. A heliostat is a set of mirror facets, called concentrator, mounted on a controllable kinematic that can track the sun and thus concentratethe incoming solar radiation onto a fixed aimpoint. Multiple heliostats combined are known as a heliostat field. To avoid dangerous structural burnups at the receiver and minimize efficiency losses, a regulated flux distribution at the receiver is desired.

2.2. Digital Twin

According to *Kuhn et al. (2017)* [13], a digital twin is a virtual representation of a model's expected behavior, that is optimized to fit measured performance data of its physical counterpart. The central motivation behind a digital twin lies in combining all relevant information about the modelled object into one entity, the digital twin. This entity can then be utilized to verify the current state of a system or predict its behavior under specific conditions.

Hence, a validated digital twin, which learned to predict the current behavior of a system, can also forecast its future behavior.

2.3. Heliostat Control

To increase the ST's efficiency, the heliostat tracking is optimized by a Concentrated Solar Power Operating System (CSP-OS) [14] as visualized in Figure 2, which uses raytracing to model the collector's expected reflection. Within the CSP-OS a digital twin (ref. section 2.2) for each heliostat is created and optimized onto performance data measurements. All heliostat digital twins combined create the heliostat field digital twin, that is applied within the raytracer to regulate the flux distribution at the receiver. Environmental impacts, such as the solar position or wind, are taken into account to predict the heliostat field's behavior.

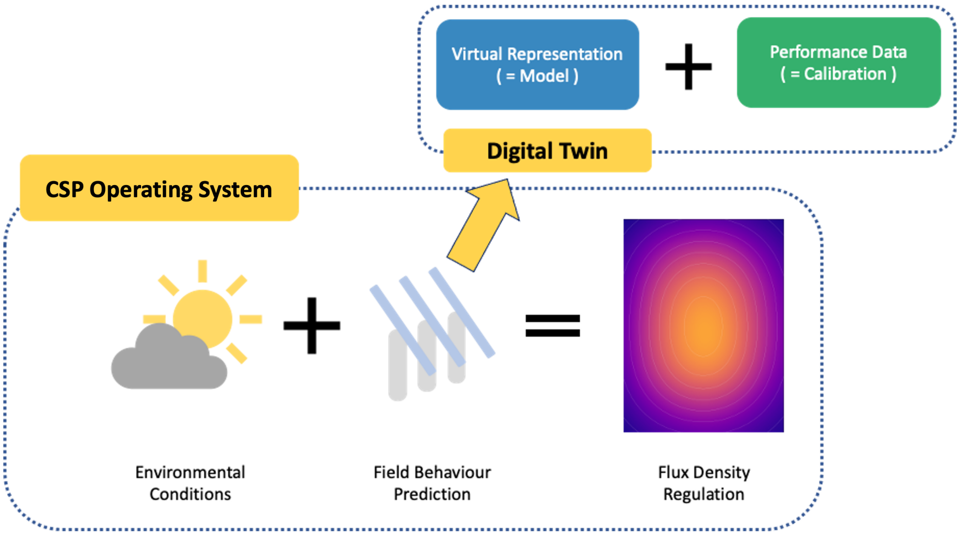

Figure 2: Representation of a Solar Tower Operating System. Environmental conditions are measured and applied to a field behavior prediction to estimate and regulate the flux density at the receiver. The prediction is obtained by a digital twin that consists of the heliostat field's model which is fitted to measured performance data.

Within CSP-OSs two different heliostat model control approaches are pursued: Open-Loop and Closed-Loop heliostat control [15]. Closed-Loop systems as shown in Figure 3 apply continuous corrections to the heliostat's actuator positions based on sensor feedback about deviations from the desired flux density distribution for each heliostat. The second approach, visualized in Figure 4, is Open-Loop control, where the digital twin's parameters are adapted on past observations. A major issue with Closed-Loop approaches is the differentation between single heliostat reflection, when multiple heliostats are focused on the receiver, which is necessary to obtain the alignment deviation and adapt the heliostat's pointing. Therefore, and for the fact, that they are easier to implement and cheaper, Open-Loop approaches are applied more often. This thesis focuses on solutions for Open-Loop systems.

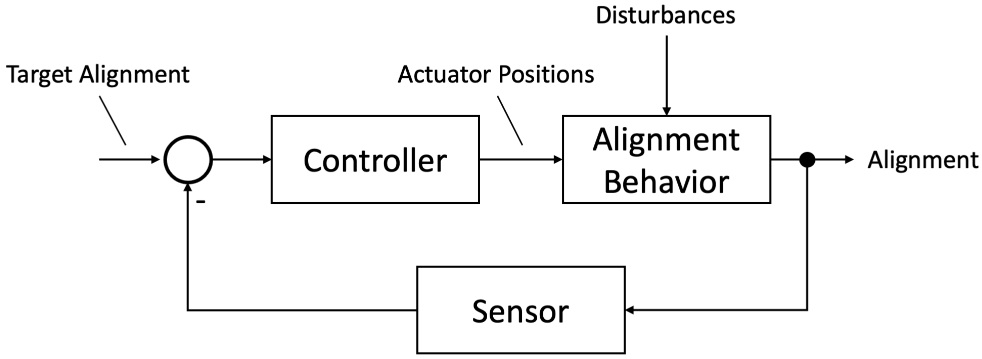

Figure 3: Solar Tower Closed-Loop Principle. Alignment deviations are measured by a sensor and passed through a controller that computes adjustments which are send to the real object and just affect its behavior.

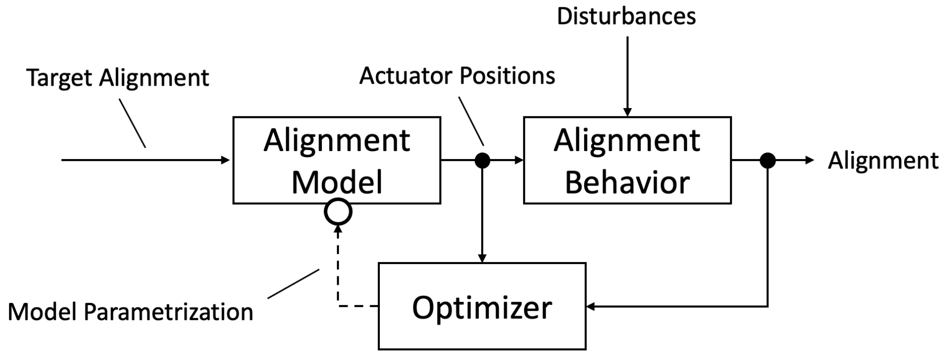

Figure 4: CSP Open-Loop Principle. Alignment deviations are measured and passed to an optimizer that creates an adapted alignment model. The adapted model is then used to predict deviations and correct them.

2.4. Heliostat Modelling

For accurate flux regulation it must be known where a heliostat's reflection is pointing and how its flux is distributed at the given receiver area for every actuator position (actuator shift via actuator steps) and vice versa. The direction where a heliostat is pointing is commonly described as its alignment which consists of the heliostat's concentrator's average normal orientation $\vec{N}_C$ and the normal's pivoting point $O_{C,Pos}$. For a known vector from pivoting point $O_{C,Pos}$ to the center of the sun $\vec{S}$ as well as the vector from pivoting point to the aimpoint $\vec{T}$ the normal N_C can be computed by the law of reflection Equation 1.

$$\vec{N}_C = \frac{1}{2} \cdot \left(\frac{\vec{S}}{\|\vec{S}\|} + \frac{\vec{T}}{\|\vec{T}\|} \right) \tag{1}$$

Heliostat alignment and flux distribution are usually modelled separately though intertwined within a heliostat's digital twin as visualized in Figure 5. To obtain the flux distribution, the concentrator's slope is modelled as a set of points and normals. Then outgoing rays from a source distribution are simulated and their reflections at the given points computed. The entirety of these points is often described as a heliostat's optical model or as used in this thesis, concentrator model. The flux distribution is given by the reflected rays' interceptions on a selected surface. The relative orientation of points and normals to each other can be determined using deflectometry [16]. However, the concentrator model's orientation within the heliostat-field depends on the heliostat's alignment. Due to their relevancy to this thesis common heliostat alignment modelling approaches are discussed in section 2.8.

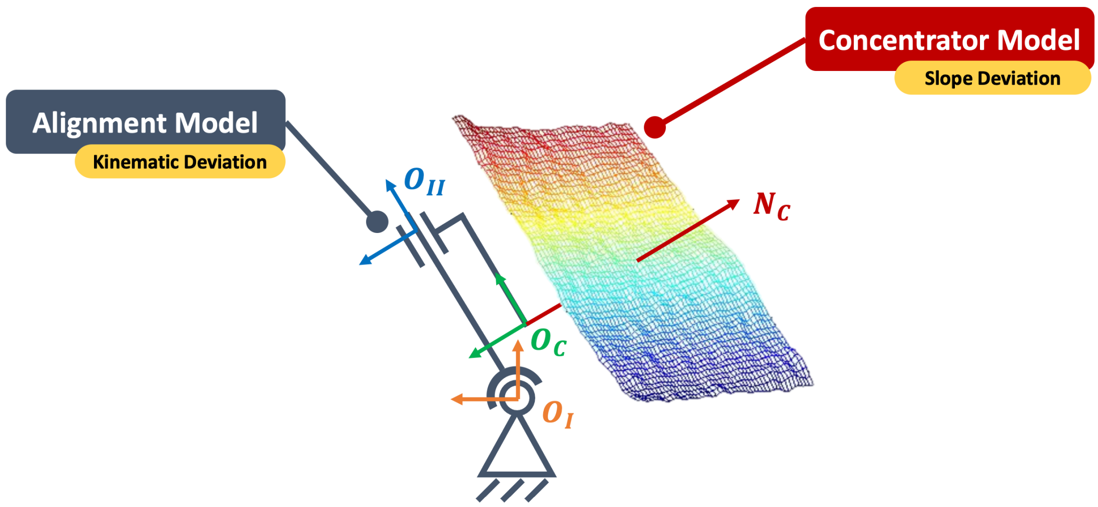

Figure 5: Heliostat Model Principle (figure adapted from [17]). The concentrator surface is modelled as a height map. The height map can then be translated and rotated in space by an alignment model. The alignment model is constructed from three coordinate systems with inter-dependent rotatory and translatory relations.

2.5. Kinematic Systems

Robotic systems often deal with the task of tool center point motion path planning based on the increment of the respective actuators. Each joint of the robotic system is therefore modelled as the origin of a coordinate system along a chain of coordinate systems. Each joint's coordinate system's rotation and location within its parent coordinate system is determined by its initial state and controlled by its respective actuator. By using extended coordinates (ref. Equation 2), the relation between two adjacent coordinate systems can be mathematically formulated as a matrix multiplication as formulated in Equation 3.

$$T = \begin{pmatrix} r_{xx} & r_{xy} & r_{xz} & t_x \\ r_{yx} & r_{yy} & r_{yz} & t_y \\ r_{zx} & r_{zy} & r_{zz} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2}$$

Each coordinate system thereby has three rotatory and three translatory degrees of freedom. Furthermore, the relation between two non-adjacent coordinate systems can be derived by combining multiple matrix multiplications along the coordinate systems chain. Kinematic modelling, however, is only physically valid if rigid bodies between each joint are assumed.

$$T_C^A = T_B^A \cdot T_C^B \tag{3}$$

2.6. Alignment Deviation

The deviation between a heliostat's alignment model's prediction and its real counterpart is commonly stated as the angle between the heliostat's predicted and targeted mirror normal in mrad (1mrad = 0.057°) and known as pointing error/deviation or alignment deviation. The latter is used further on. The cathesian deviation at the target for the same alignment deviation angle, thus growths by the distance between heliostat and target. For larger heliostat fields, which commonly exceed a receiver heliostat distance of 1000m, even smallest alignment deviations lead to heliostats potentially missing the receiver as demonstrated in Figure 6.

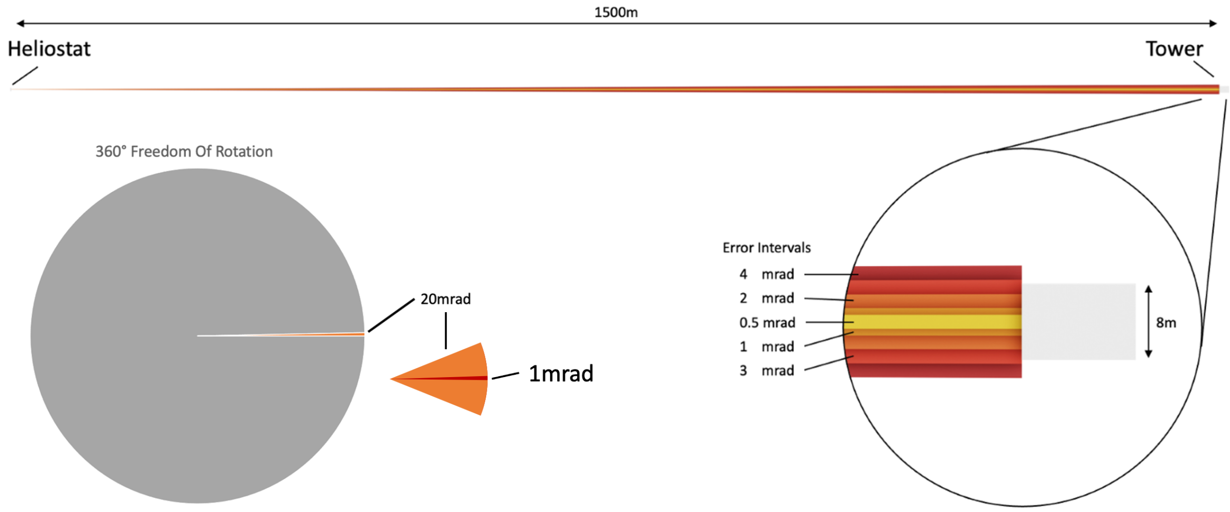

Figure 6: Alignment deviation effects at a distance of 1500m for pointing errors between 0-4mrad and visualization of 1mrad compared to a full circle.

Furthermore, the flux density $\dot{Q}_{abs}$ at a receiver for a given heliostat field flux concentration $\dot{Q}_{conc}$ decreases depending on the receiver area A_{rec} (ref. Equation 4). The plant efficiency however increases with the achievable process temperature and thus $\dot{Q}_{abs}$ [comp. X] and for some thermo-chemical applications temperatures > 1400°C are required.

$$\dot{Q}_{abs} = \dot{Q}_{conv} - \varepsilon A_{rec} \sigma T_{rec}^4 - \dot{Q}_{conv} \tag{4}$$

Therefore, small alignment deviations in the vicinity of 1mrad are desirable. Figure 6 further illustrates 1mrad compared to a full circle. The figure not only demonstrates the required prediction accuracy but also the precision with which the alignment deviation must be measured to train the heliostat model's parametrization to fit its real counterpart.

2.7. Camera Target Method and Heliostat Calibration

Heliostat alignment measurement tools are commonly known as BCS as they usually observe a heliostat's beam reflection for a known source position and thus deduct the heliostat's alignment. An overview of the most common BCS is given by *Sattler et al. (2020)* [8]. Of relevancy to this thesis is the Camera Target Method, shown in Figure 7, which is the current industry standard.

The Camera Target Method uses a Lambertian target plane that is digitized via a fixed camera. For a given source position, a heliostat's projected reflection onto the target equals the heliostat's flux density and can be detected by the camera as light intensity upon the target. Commonly the flux density is then reduced to the flux density center. The same process is repeated in a simulated environment using the heliostat's digital twin. Then the deviation between predicted and real flux density center can be extracted. The procedure can be repeated for multiple source positions and the digital twin's model parametrization is fitted to predict the real-world measurements as precisely as possible.

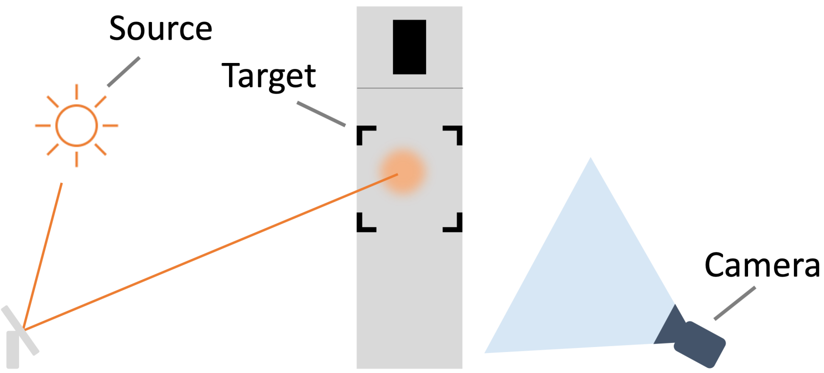

Figure 7: Visualization of the Camera Target Method. The source (e.g. sun) is reflected by a heliostat onto a target. A camera is used to determine the reflection's distribution upon the target and derive a deviation from desired to actual reflection. The deviation is than used to calibrate the heliostat's digital twin.

Most heliostat fields that implement the Camera Target Method have one target and thus generally can collect performance data for one heliostat at a time only, to ensure that the measured flux density is assigned to the correct heliostat. This method is furthermore limited by its target size. Alignment deviations that exceed the calibration target area cannot be detected automatically and must be corrected either manually or via an search path algorithm, which both are time-intensive procedures. Therefore, a rather optimistic average calibration point collection speed of 10s per point for this method is stated by *Sattler et al. (2020)* [8], whereby *Smith et al. (2014)* [18] states 30s for a similar technique.

The process of collecting heliostat performance data is often called heliostat calibration. This is misleading as in CSP terms heliostat calibration is not ambiguously defined. For this thesis heliostat calibration is assumed to describe the entire process from performance data gathering and heliostat modelling to digital twin training the heliostat's alignment. Technically, concentrator model training is part of the calibration process as well but excluded from the scope of this thesis as its focus lies on digital twin alignment modelling.

2.8. Heliostat Alignment Modelling Approaches

Physically, a heliostat's alignment function depends on multiple factors as can be seen in Figure 8, further called inputs. The most relevant inputs to a heliostat's alignment are the actuator positions ϑ, τ as these are the variables that can be manipulated by the heliostat's control loop. The actuator positions directly control the heliostat's kinematic and therefore its collector's orientation. All other inputs act as disturbances to the control loop. Further notable inputs are environmental impacts on the heliostat such as wind or temperature, or wear which can be approximated as function over time. Additionally, some effects are of irregular occurrence (e.g. mirror damage due to hail) and thus generally can't be estimated a priori but included into future predictions after their appearance.

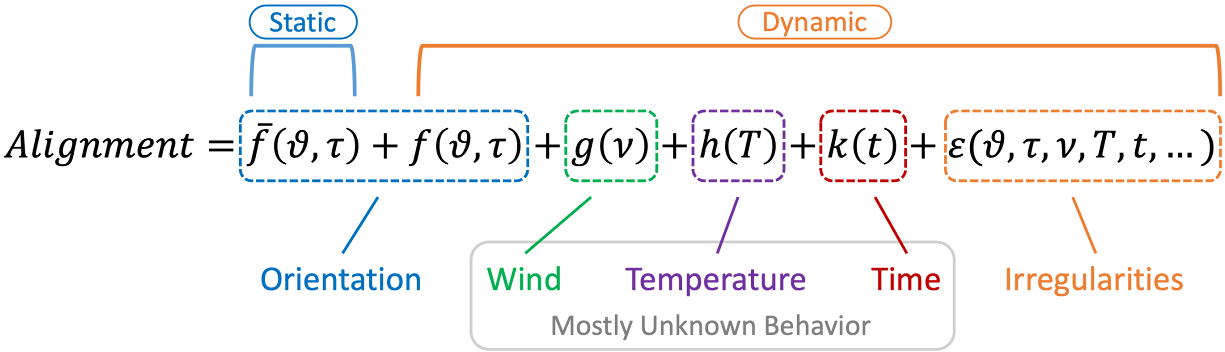

Figure 8: Impacts on a heliostat's alignment devided into categories of orientation and environmental impacts. The impacts can be split into static behavior that can be modelled by constant parameters and dynamic behavior that depends on the heliostat's current state.

Alignment inputs can generally be divided by their impact type into two categories: static and dynamic behavior. Static behavior are constant offsets from an ideal assumption that are introduced during the manufacturing and installation of the heliostat and therefore could be measured and accounted for once a heliostat is installed. Dynamic inputs in contrast depend on a heliostat's current state such as its surrounding environmental condition or the heliostat's structural bending due to load in different orientations. Besides its orientation part, alignment inputs are thus mostly dynamic. The orientation depends both on manufacturing and installation deviations as well as structural bending and thus can be split into a static and dynamic part. Physically, the orientation's static part equals a kinematic behavior that assumes rigid-body behavior but allows for parameter deviations.

Rigid-body behavior can be modelled by kinematics and thus is the standard approach of heliostat alignment modelling, but therefore only accounts for static heliostat behavior. Current research shifts to focus on alignment models that include dynamic behavior. All alignment modelling approaches can be divided into three categories which are explained within the next sections.

2.8.1. Rigid-Body Heliostat Alignment Modelling

Historically, the approach first suggested by *Baheti et al. (1980)* [19] and visualized in Figure 9 is the most implemented one [20, 21, 22, 23, 24, 25, 18, 26, 27, 28]. The heliostat alignment model is derived from an investigation of the heliostat's kinematics and actuator geometry.

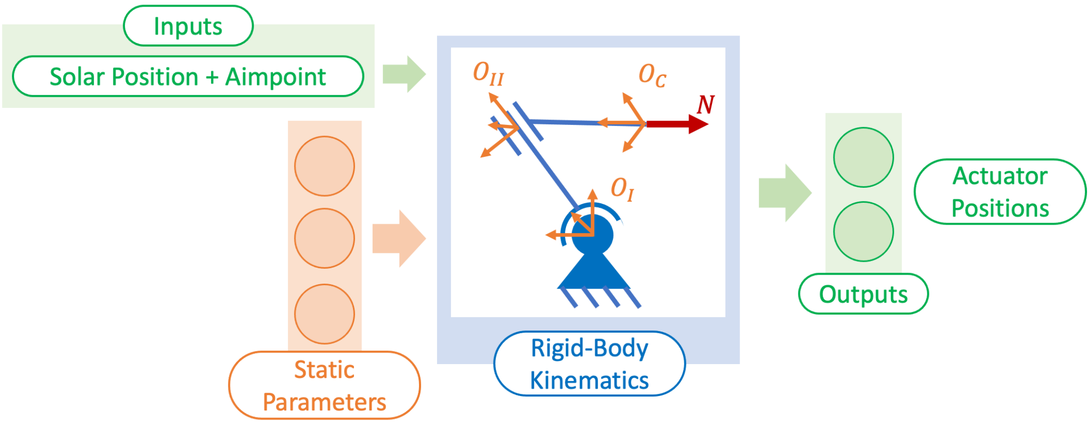

Figure 9: Solar positions and aimpoints are passed to a rigid-body kinematic model. As a result, actuator positions are predicted by the model based on the model's static parameters. The static parameters can be adapted to change the model's behavior.

A heliostat's kinematic commonly consists of two rotatory joints with one degree of freedom which are connected by a rigid strut and modelled as rotation matrices $O_{I,Rot}$, $O_{II,Rot}$ and a constant vector $\overrightarrow{O_I O_{II}}$ in between. The concentrator is connected to the second joint by another rigid strut which again is modelled as a fixed vector $\overrightarrow{O_{II} O_C}$. To rasie the heliostat's pivoting point and allow for larger concentrators, the first joint is mounted on top of a pole. While modelling, the pole is neglected and can be condensed to the first joint's origin $O_{I,Pos}$. The joints' origins are chosen to equal their pivoting points. Via a combination of translations and rotations the heliostat's alignment can be determined relative to the first joint's origin and then translated into the global heliostat-field coordinate system. Due to its geometric nature this approach is also known as geometry model, kinematic model. Within this work the term rigid-body model is used to differentiate it from the other alignment modelling approaches.

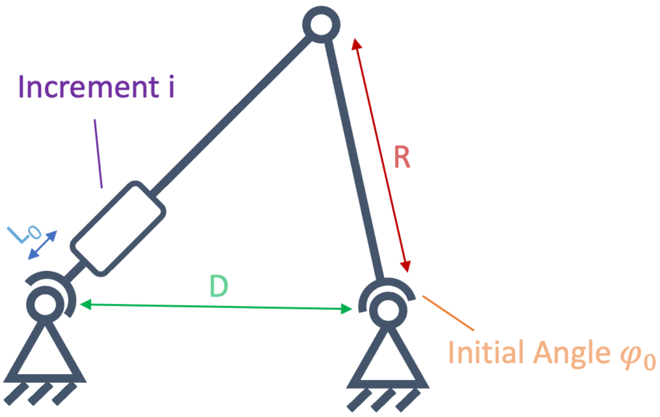

Figure 10: Simplified model of a heliostat linear actuator, where the heliostat's concentrator is rotated around a pivoting point via a change in length of one of the actuator's struts.

Each joint's rotation is controlled by either a rotatory or linear actuator. The common case is linear actuators. If the linear actuator is assumed to be ideally orientated within a plane, the joint's rotation in dependance of the actuator's length can be derived from the law of cosine (ref. Figure 10, Equation 5, Equation 6). Therefore, the initial actuator length L_0 and distance D between actuator origin and joint pivoting point as well as the joint's pivoting radius R must be known. Of further relevance is the actuators gear ratio i from steps to length difference and initial angle offset φ_0.

$$L(s) = i \cdot s + L_0 \tag{5}$$

$$\varphi(s) = \arccos \frac{D^2 + R^2 - L(s)^2}{2DR} + \varphi_0 \tag{6}$$

This approach models a heliostat's ideal behavior and thus is the most stable one as no training data must be gathered for the model to reasonably predict a heliostat's alignment. To better approximate a real heliostat's behavior, parameter optimization is performed on training data and can be formulized as Equation 7. These parameters $\bar{W}$ are constants within the model and thus can only fit a heliostat's average deviation from its ideal behavior. Therefore, this approach is limited to reducing a heliostat's, manufacturing and installation deviations with limited capabilities to reduce the impact of dynamic behavior and environmental factors or wear over time.

$$Alignment = f^* \left(\vartheta, \tau, \bar{W}\right) \tag{7}$$

Baheti et al. (1980) [19] introduced six disturbances: three rotations around the east, north and up axis, one gear ratio disturbance for each of the two actuators and a bias due to measurement errors of the elevation angle. *Stone et al. (1986)* [20] extended the number of disturbance corrections to eight. *Smith et al. (2014)* [18] described these eight parameters as "pedestal tilt about the east axis", "pedestal tilt about the north axis", "azimuthal reference bias", "elevation reference bias", "azimuth linear error", "elevation linear error", "drive-axis non-orthogonaility" and "boresight error". *Freeman et al. (2017)* [26] first applied Denavit-Hartenberg notation and thus extended coordinates to alignment modelling.

2.8.2. Neural Network Heliostat Alignment Modelling

The most general approach, visualized in, is using a neural network which predicts the alignment. This approach can learn to model a heliostat's complete alignment behavior, including dynamic impacts such as wind and thus promises high prediction accuracies. Furthermore, no prior analysis of the heliostat's kinematics is required. Possible implementations were thus pursued by [29, 30, 31, 11, 9]. A major drawback to this approach is the prediction's complexity. This leads to high amounts of required training data [11] to successfully modify the networks parametrization. *Pargmann et al. (2021)* [11] therefore suggested pretraining the Neural Networks on simulated data, to preconfigure the neurons' linkage and thus reduce the amount of required training data.

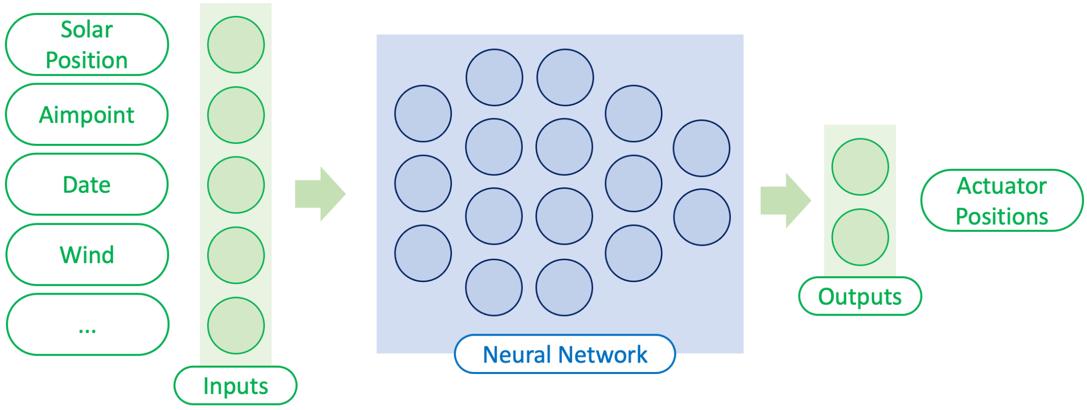

Figure 11: Multiple inputs are passed through a network of neurons to predict output values.

2.9. Alignment Duality Problem

A further aspect to consider when modelling heliostats is the alignment model's duality problem. The above introduce alignment modelling approaches all take solar positions and aimpoints as inputs and predict the heliostat's corresponding actuator positions. This task is mandatory for heliostat field control. For flux distribution prediction using ray-tracing the inverse direction of computing alignments from actuator positions is required.

For the rigid-body kinematics approach this can be solved iteratively as shown in Figure 12. First arbitrary actuator positions are chosen as an initial guess. From this initial guess a concentrator pivot point can be derived and a targeted concentrator normal computed that would reflect light from the given source position to the desired aimpoint. This targeted concentrator normal does not necessary conform with the one predicted by the alignment model for the actuator positions guess. In that case inverse kinematics can be solved either iteratively or analytically to derive actuator positions that fit the targeted concentrator normal. From the obtained actuator positions a new concentrator pivot point and thus concentrator normal can be derived and thus a new targeted concentrator normal. This process can be repeated until it converges to a combination of concentrator pivot point and normal that reflect light from the source position to the aimpoint within a preset tolerance.

The earlier introduced Neural Network alignment model approach skips computing the concentrator pivot point and normal. This can be mitigated using an assumed concentrator pivot point and computing the concentrator normal from source position and aimpoint for this fixed pivot point. This allows for adapting the neural network to take the concentrator normal as input. Besides this approach being only an approximation to the real-world behavior of the heliostat, neural networks generally are not invertible [32] and thus a concentrator normal could only be derived iteratively from given actuator positions.

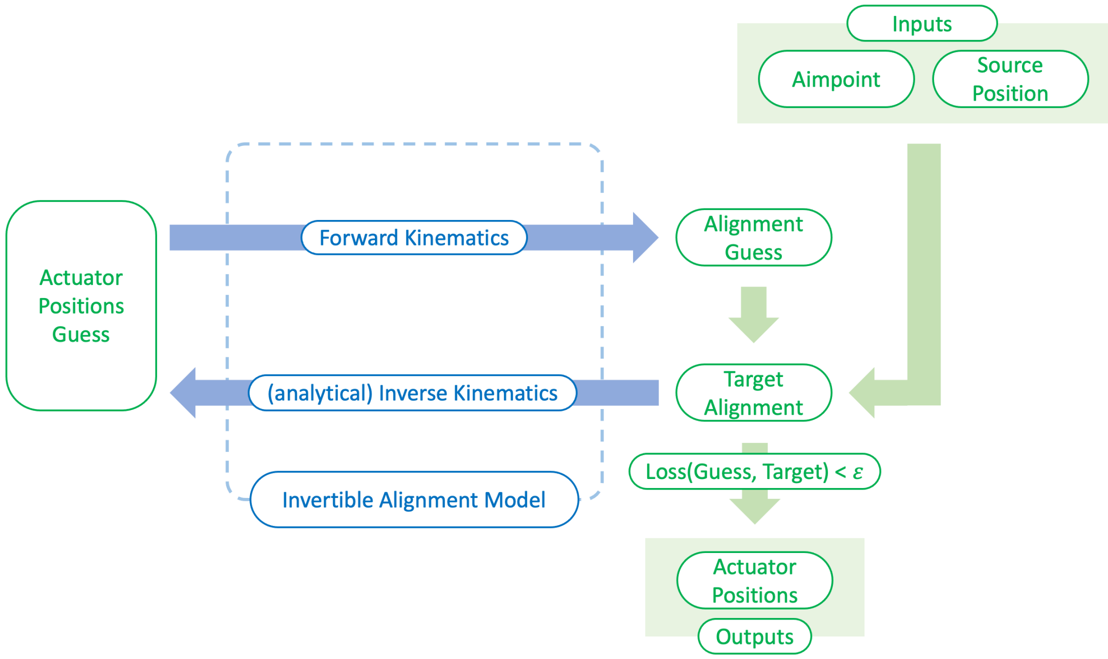

Figure 12: A heliostat alignment is computed by passing arbitrary actuator positions to the alignment model. Based on a given aimpoint and source position and the initial alignment's concentrator center, a target alignment is computed that follows the law of reflection. An updated actuator positions guess can be obtained via the model's inverse direction. This process is repeated by computing a new alignment from the updated acutator positions until the alingment guess is sufficiently close to the target alignment that results from the law of reflection.

2.10. Differentiable Raytracing For Solar Tower Plants

Pargmann et al. (2023) [12] developed a ray tracer for the application within STs that is designed to enable automatic differentiation along the ray tracing process using PyTorch [33]. Automatic differentiation refers to a technique where a gradient graph is constructed by adding the gradient of each node. Using the gradient graph the gradient for a node's children can be extracted using the chain rule. Every equation is split into its elementary computations which are then added to the gradient graph. Especially a gradient of a loss function can thus be passed along the graph. Using machine learning procedures such as gradient descent, the gradient graph's leaves can be optimized to minimize the given loss function.

In the context of a differentiable ray tracer the loss function is the quantified deviation of a measured flux density to the predicted one. As the flux density prediction depends on the heliostat model's parameters the loss gradient for each model parameter can be computed and used to fit the model to predict the actual flux density with as little deviation as possible. Furthermore, the loss can be constructed from multiple sets of real and predicted flux densities. Another benefit is the possibility of multiple additional loss terms such as a direct alignment deviation loss for the investigated heliostat.

The loss gradient nature of this ray tracer also allows to optimize a heliostat' alignment and concentrator model's parametrization at the same time. Additionally, the information loss that happens when the flux density is reduced to its density center can thus be avoided. Another benefit of the ray tracer is its architecture's similarity to other neural network training approaches. Therefore, the heliostat model or parts of it can be exchanged by neural networks. Prior to this thesis however, only simplified heliostat alignment models and no neural networks were tested within DIRC.

2.11. Deep Imbalanced Regression

Yang et al. (2021) [34] introduced Deep Imbalanced Regression (DIR) as learning to predict continuous patterns from a dataset where some patterns are over-represented while others might be completely missing. The loss term that is responsible for updating a model's parameters usually is a metric (e.g. Mean Squared Error also called L2 (MSE)) over each data point's prediction error that is not sensitive for the data's distribution and thus is biased towards patterns of higher frequency [35].

In classification tasks, classes are therefore divided into majority and minority classes, where majority classes have higher sampling frequencies than minority classes and thus are better learned. A classification dataset imbalance indicates few majority and multiple minority classes. This approach can't be applied to continues patterns. To interpret a regression dataset's balance the definition of a distance function between patterns is suggested. Therefore, majority patterns are those of low distance and minority patterns those of high distance. The next two sections focus on summarizing common distance functions and imbalance mitigation techniques.

2.11.1. Distance Functions

When computing distances within datasets three aspects are of relevance [36]: the distance notion, dimensional scale, and distance prototype. The most popular distance notions, visualized in Figure 13, are Euclidean, Angular and Manhatten distance, whereby the Euclidean and Manhatten distance are special cases of the Minkowski distance. Choosing a distance notion depends on the data's equidistance behavior.

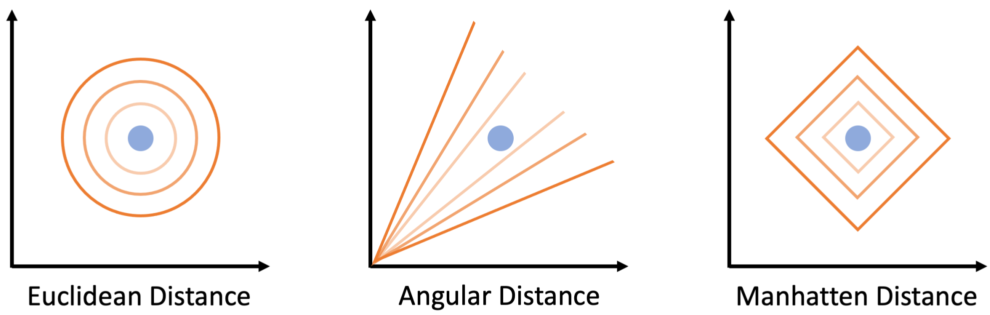

Figure 13: Distance notions (orange) visualized for an exemplatory data point (blue). High distances are indicated without transparency, low distances with high transparency. (adapted from [36])

When comparing data with a multi-dimensional feature-space dimensional scaling becomes mandatory, as the magnitude of impact can vary between different feature-axes units. Within this thesis distances between features of equal units are analyzed and thus scaling techniques are left for future discussions.

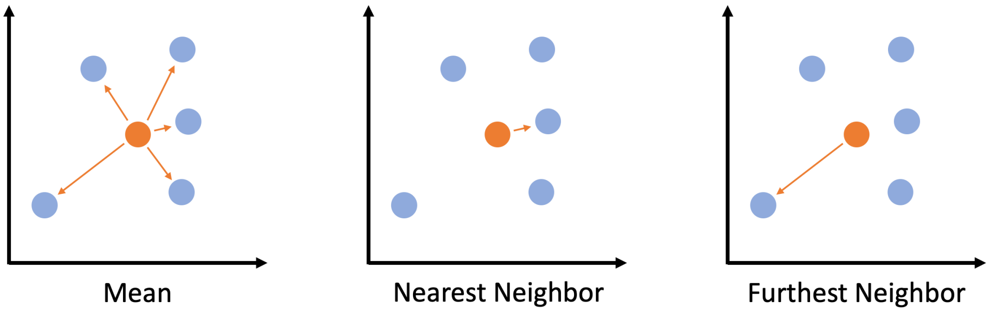

Figure 14: Distance prototypes between an examplatory data point (orange) and a dataset (blue). Relevant distances for each prototype are indicated by an orange arrow. (adapted from [36])

The distance prototype refers to the point where a distance is pointing to. Common distance prototypes (ref. Figure 14) are Mean, Nearest and Furthest Neighbor. Mean Prototype averages a points distance to all other points within a dataset. Nearest Neighbor Prototype only considers the smallest distance to any other data point within a dataset and Furthest Neighbor respectively the largest distance. Special cases of the Nearest Neighbor Prototype are k-Nearest Neighbor Prototype and k-Mean Nearest Neighbor Prototype. k-Nearest Neighbor considers the sum of distances over the k nearest neighboring data points. Alternatively, the mean over the k nearest neighbor distances can be computed, thus resulting in the k-Mean Nearest Neighbor Prototype.

2.12. Machine Learning Training Techniques

In supervised machine learning a model is trained to predict a given set of patterns by learning from training data that samples the patterns. Besides a machine learning model's architecture and optimizer there exist several techniques to optimize a model's training capacity by adapting its hyper parameters. Hyper parameters are values that control the model training behavior, and which are set outside the training process. Those techniques relevant to this thesis are summarized below and serve either to prevent vanishing gradients or overfitting.

Vanishing gradients occur when the model's parameters' arrangement diminishes the parameters' gradients with each gradient step and thus prevent the model from further learning before an optimal solution is found. Overfitting instead describes a model's state where it learned to correctly predict its training data but only found a local minimum and thus is not capable of abstracting its learnings onto prior unknown tasks.

2.12.1. Learning Rate and Early Stopping

The first technique to be mentioned is the adaptation of the applied learning rate. An epoch is the training step between two gradient backward propagations and the learning rate controls how much the parameters are adapted. Either the learning rate's initial value can be changed or a learning rate decay over the number of training epochs is introduced. Small learning rates take many epochs to train the model and might get stuck in local minima. Too large learning rates in contrast might lead to the training fluctuating without improving, when the optimal solution would require smaller parameter adjustments.

2.12.2. Input Normalization and Encoding

Another technique that supports the model's learning process is to normalize all input parameters, not only by value but also to equalize the gradients order of magnitude. A further approach is input multiplication using Fourier feature mapping as suggested by **??**. This approach was found to significantly improve a neural network's performance for low-dimensional regression tasks, where high-frequency features within the input data were not noticed by the network until explicitly been introduced as additional inputs. These additional inputs are obtained by splitting each input parameter's value into multiple inputs, one each for a different frequency band. This can be achieved by using the Fourier transform. The resulting positional encoding can be obtained from Equation 8. Where $\gamma(p)$ are the resulting parameters derived from the original parameter p.

$$\gamma(p) = \left(\sin 2^0 \pi p, \cos 2^0 \pi p, ..., \sin 2^{L-1} \pi p, \cos 2^{L-1} \pi p\right) \tag{8}$$

2.12.3. Dataset Splitting

A good practice in machine learning is splitting a given dataset into training, validation and testing data. Testing data is data the model has not seen before and acts as a benchmark for the model's abstraction capabilities. To ensure valid benchmarking results the testing data should be uncorrelated from data that was used for training or validation. Furthermore, testing data should cover all patterns of interest. Generally, a model's performance is evaluated by regarding the mean prediction deviation on the testing dataset. To ensure unbiased results the testing dataset should therefore further be balanced.

Training data are the information a model is tuned to predict, by computing a training loss and adjusting the model's parameters via back-propagation. Hence, models tend to overfit onto the training data. To overcome the overfitting issues training data can be split into training and validation data. Validation data is excluded from the training process but used as an indicator on the model's abstraction capabilities during the training process. Therefore, commonly model parametrization's are stored for the epoch that best performed on the validation data. After the training is completed, the best performing parametrization is restored.

2.12.4. Weight Decay Regularization

In machine learning the complexity behind the patterns to be learned is generally unknown. Therefore, the model complexity should be at least as high as the pattern complexity. Too high model complexities can lead to overfitting where the learned training data is predicted correctly but the understood pattern deviates from the targeted one. One technique to reduce overfitting is weight decay regularization.

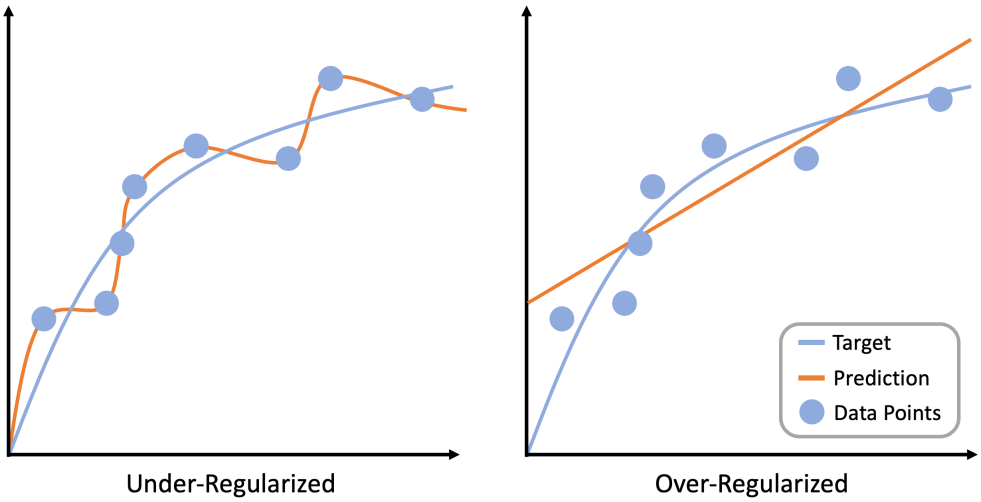

Figure 15: Regularization Effects (adatped from [37]). Both plots show a model's learned function (orange) after the model was trained on data points (blue) with measurement errors. The training target function is shown in blue. The left plot visualizes under-regularization, resulting in a function with high variance that is capable to predict all points correctly. The right plot instead shows over-regularization, resulting in a function with low variance, that thus fails to correctly learn the targeted behavior.

Weight decay regularization [37] penalizes a model's complexity and thus pushes it to the minimum amount of required complexity. [Figure X] visualizes the effects of a regularized model's prediction (left) against a prediction that is too complex (right). To reduce a model's complexity an additional loss term beyond the model's performance loss is introduced. This loss is the sum over each model parameter's weight, whereby the weight can either be defined as the parameter's absolute value in which case the regularization is known as L1-Regularization or as the parameter's squared value which is called L2-Regularization.

2.13. Neural Networks

A universal function is a model that can be fitted to learn to predict any pattern. Neural Networks are one of the most popular universal functions. Generally, a Neural Network's architecture mimics the principles found in a brain. A directed acyclic graph is constructed where all source leaves are inputs to the universal function and all sink leaves are outputs. Due to its acyclic behavior the term Feed-Forward Neural Network is often used. The graph's nodes are called neurons and clustered in layers. Besides an input and output layer multiple layers can be stacked as a sequence of layers, whereby all neurons within one layer are connected to all neurons within adjacent layers. Each arc between two neurons is weighted by the impact of the first neuron on the second neuron. Each neuron's value thus is the sum over all adjacent neurons weighted by the incoming arcs. Generally, small amounts of neurons are required to create a universal function, but deep neural networks with multiple layers were found to yield best prediction performances. Explaining all aspects of Neural Networks exceeds the scope of this thesis, therefore the two most relevant properties to this thesis are introduced within the next sections.

2.13.1. Self Normalizing Neural Networks

Model prediction tasks can be split into two categories, classification and regression. Classification tasks try to assign given input parameters to an a priori known set of classes. Regression instead tries to predict a set of output parameters based on its inputs. While a classifications model's output is limited to distributing percentages between its known classes, a regression's outputs generally are unbounded. This, however, makes it harder for models to learn to predicted outputs that are naturally bounded or have unique solutions for only a given range of outputs (e.g. rotations). Approaches that use bounded regression models, come at the expense of gradient clipping at their boundaries, which can lead to vanishing gradients. To circumvent these issues Self Normalizing Networks (SNNs) [38] can be used. The basic principle behind SNNs is that the prediction by construction of the neurons strives for a prediction mean value around zero with a standard distribution value of one. While it is possible for a SNN to learn exceed this prediction range, all exceeding values experience a pull towards the targeted range of prediction. Thus, prediction bounds can be accomplished without the issue of gradient clipping.

3. Related Work

Due to its high impact on CSP plant efficiency, multiple advances in heliostat alignment modelling and optimization have been recently published. Relevant to this thesis are two aspects of these publications. The first aspect is an analysis of the used dataset splitting into training, validation and testing data. This is relevant because of the chosen split being known [9] to have an impact on the achieved prediction performances and thus is necessary when comparing prediction capabilities. The second aspect is the publications' result presentation, which should act as a basis for this work's result presentation. All publications were found using Google Scholar and the following search keys: "Open Loop Heliostat Control", "Heliostat Calibration", "Heliostat Tracking", "Solar Tracking Neural Network".

3.1. Dataset Selection

Khalsa et al. (2011) [39] trained their model on data collected between 12:30pm-4:40pm on June 16 and 9:30am and 3:55pm on June 17, 2011 at an interval of 15-30 minutes and tested on data of July 15, 2011 12:53pm to 4:30pm at an interval of 3 to 4 minutes. The authors also stated environmental conditions during their measurements. A training dataset size of (250min + 385min) / (15-30min) equaling roughly 30 data points. Figure 16 gives a rough estimation of the chosen training and testing data points plotted as solar azimuth and elevation coordinates. As both training and testing data were taken close to the summer solstice and at the same time of day, the solar positions might be too similar to assume successful generalized predictions.

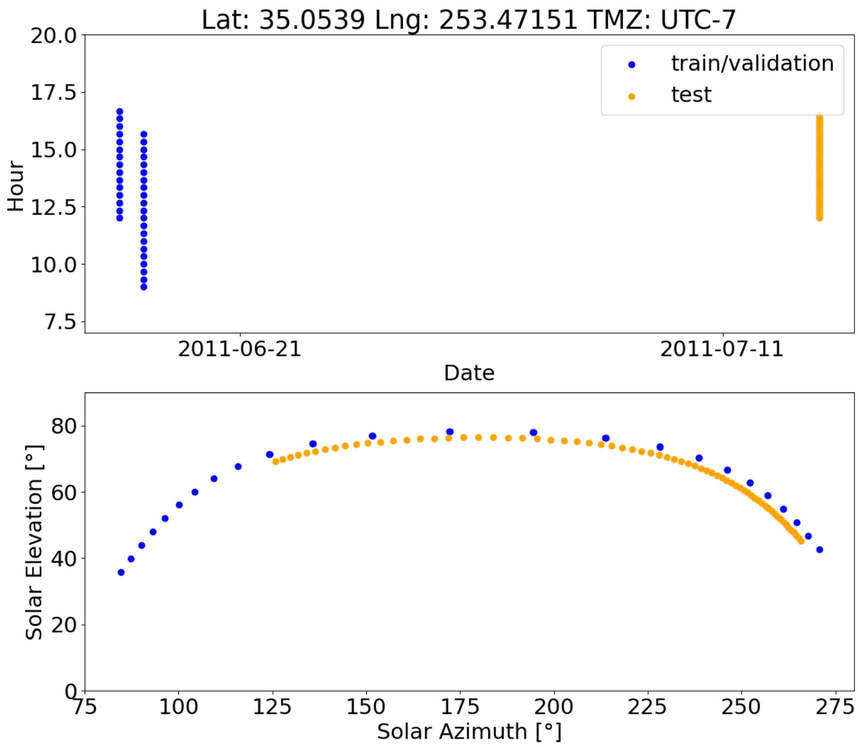

Figure 16: Approximated data distribution of the dataset used in [39]. While apart in time, both training (blue) and testing data (orange), show resembling solar positions.

A more significant distinction between training and testing data was presented by *Pargmann et al. (2021)* [11] by selecting training data at the end of July to the mid of August and testing on data starting at the end of August to early February. An estimation of the applied data splitting is given in Figure 17. This led to high solar elevation deviations between training and testing data. A further improvement might be additional distinguishing over the data points' azimuth.

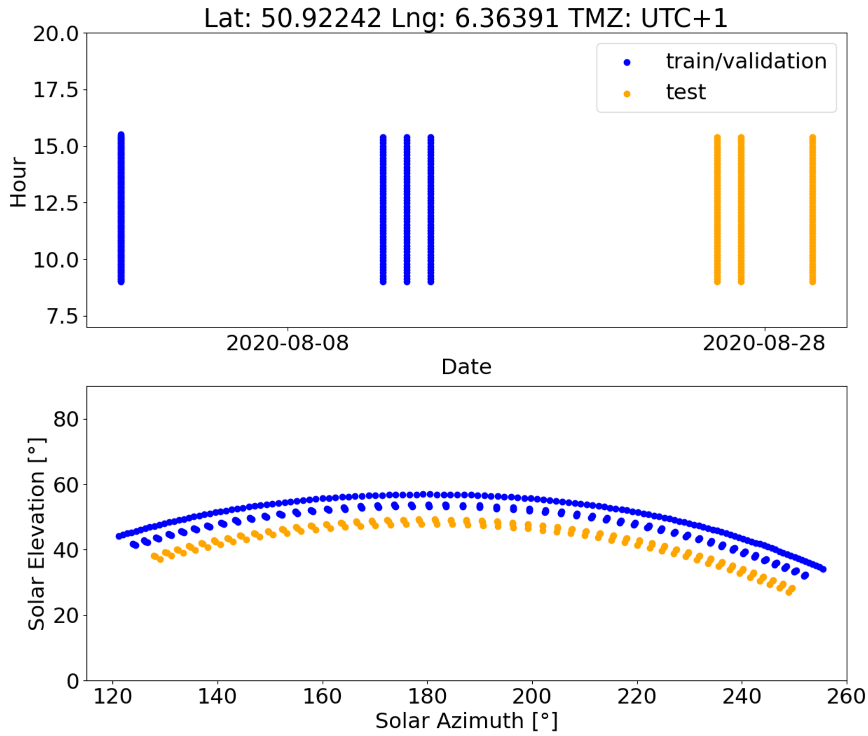

Figure 17: Approximated data distribution used in [11]. Even though the test dataset (orange) was chosen to be more than a month apart from the first training data points (blue), the resulting solar position distributions show only small differences.

A major time difference of six months was evaluated by *Smith et al. (2014)* [18]. Training data was collected between August 9-27 2012, October 22-29 2012 and February 4-5 2013. Two testing campaigns were completed between October 30 to November 1 2012 and February 12-27 2013. Data per heliostat was collected every other day at 1h intervals. As can be seen in Figure 18 the timely difference between October and February still results in similar solar angles.

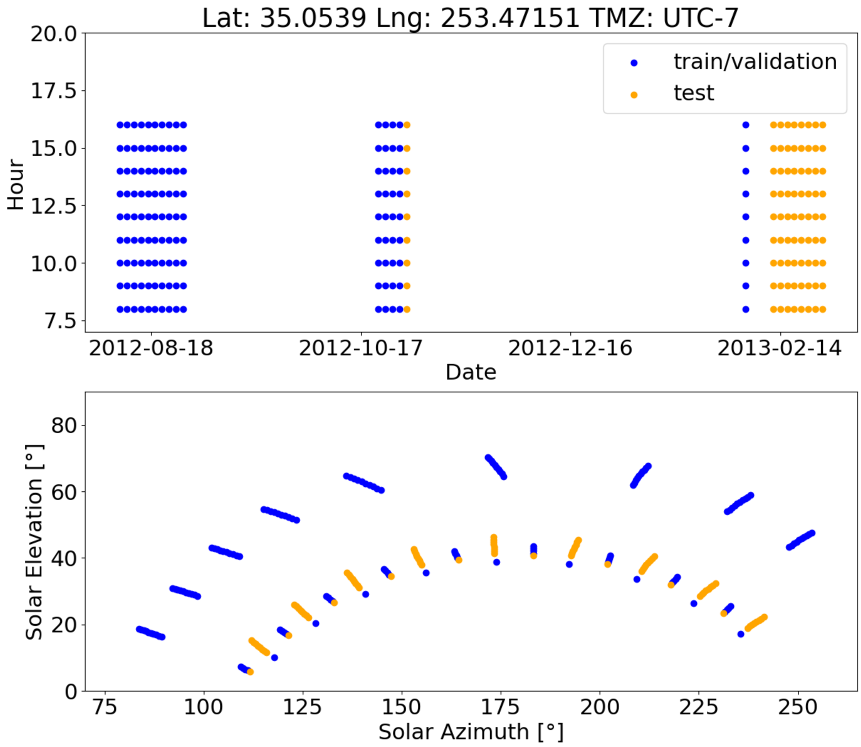

Figure 18: Approximated data distribution used in [18]. Due to the solar positions' resemblance in October and February, highly similar data distributions where obtained for training (blue) and testing data (orange), even though the datasets were strictly separated in time.

3.2. Model Performance Evaluation

Al-Rousan et al. (2020) [31] plotted predicted heliostat orientations in degrees over number of samples in comparison to ideal orientations as shown in Figure 19. This plot excels at visualizing the learned prediction capabilities. The targeted prediction accuracies of 1mrad, (0.057°), can't be visualized within a scale of 140 degrees though, making this plot generally not applicable for this thesis.

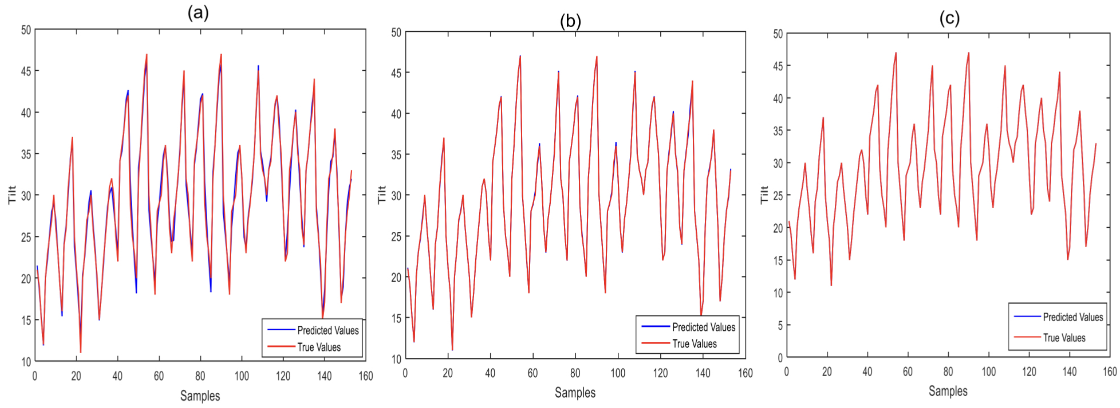

Figure 19: Result visualization copied from [31], showing the predicted heliostat behavior on different amounts of tracking errors (red) compared to the target behavior blue. Due to the high precisions required in heliostat tracking, little differences can be observed.

Smith et al. (2014) [18] stated all prediction performances together with untrained model performances as shown in Figure 20. Furthermore, the average of tracking improvements due to model prediction were averaged over 216 different heliostats.

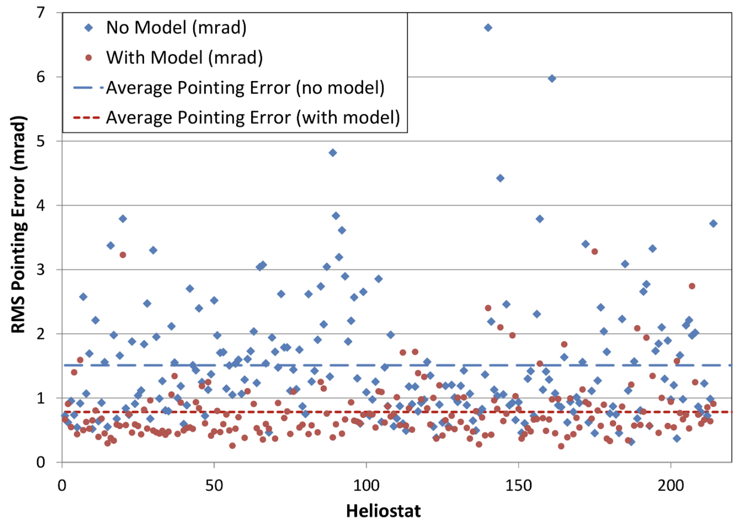

Figure 20: Result visualization copied from [18], showing the achieved prediction accuracies before (blue) and after model training (red) as pointing errors for multiple heliostats. Furthermore, the achieved average pointing errors with (red) and without (blue) modelling are indicated as dotted lines.

Pargmann et al. (2021) [11] indicated prediction deviations over time and gave a further plot of prediction deviations over number of data points as visualized in Figure 21. A similar approach was chosen by *Sarr et al. (2023)* [9] and *Malan et al. (2014)* [40].

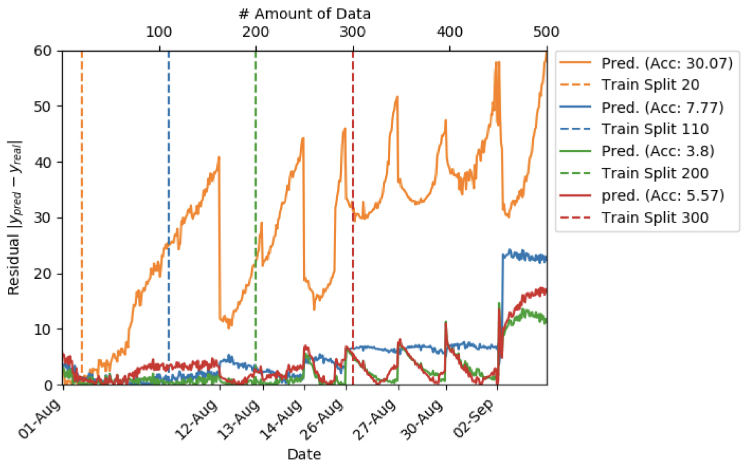

Figure 21: Result visualization copied from [11], showing the performance over time of one heliostat for different train/test splits. The splits are applied by day and indicated as dashed lines.

4. Dataset Distance Metric

As summarized in section 3.1, most current publications focus on time as a major influence on a heliostat's behavior. From a technical point of view the most major impact is the heliostat's alignment. The alignment further depends on the vector between solar position and aimpoint. While time is correlated to the solar position, their relationship is not unique in the sense that two points in time can be separated by a timespan of multiple days but still have similar solar positions. Therefore, sampling by time does not guarantee dataset balancing (e.g. bias towards morning data) and thus can even reduce the trained model's abstraction capabilities. Figure 22 visualizes the solar positions relative to the heliostat field at STJ for the first day of each month throughout 2022. As can be seen, the solar positions for each month in the first half of the year overlap solar positions in the second half of the year.

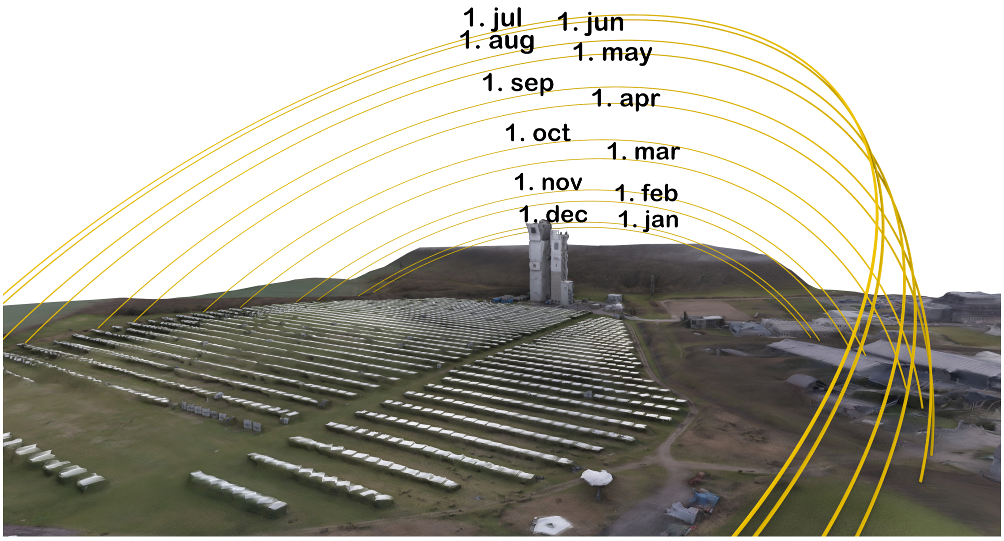

Figure 22: Solar positions at Solar Tower Jülich relative to a fixed point within the heliostat field for the first day of each month in 2022 shown as yellow arcs.

The following subsections deduce a spatial distance metric between data points' solar positions. Based on the introduced distance metric an analysis for a dataset's coverage of a heliostat's behavior is derived. Furthermore, two algorithms for dataset sampling are suggested.

4.1. Assumptions

To understand the motivation behind the new metric's development, a basis of three assumptions about a heliostat's behavior is introduced within this section.

4.1.1. Behavior Solar Position Correlation

The first assumption is that a heliostat's behavior mostly depends on its alignment (ref. section 2.4). This can be explained by the structural-bending and gear-play correlating to the heliostat's orientation. Furthermore, the alignment again depends on the vector between heliostat concentrator and solar position, due to the heliostat's alignment being adjusted to reflect the emitted solar rays to an aimpoint. Therefore, a corrleation between heliostat behavior and solar position is assumed.

4.1.2. Model Performance Behavior Distance Correlation

The second assumption about a heliostat's behavior is that no other effect but the distance between heliostat alignments is relevant to the prediction accuracy. This assumption enables exploring the correlation between model prediction accuracy and dataset distribution with limited knowledge of all correlated aspects. The motivation behind this assumptions is the idea that alignment changes relate to dynamic effects onto the heliostat's kinematics and therefore change a heliostat's behavior. Heliostat's that don't change their alignment further are expected to obtain their current behavior over time.

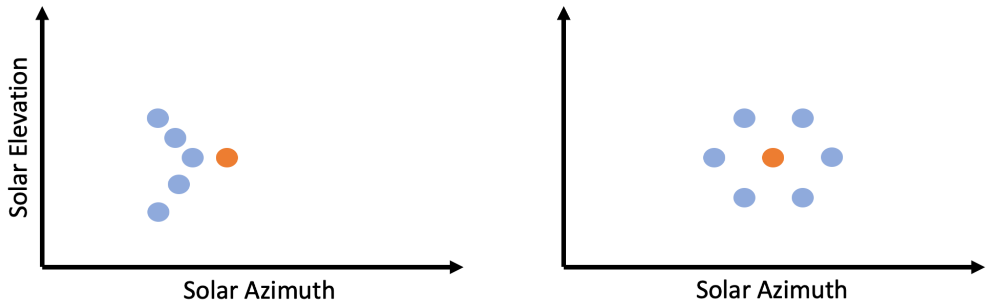

Figure 23: Close Extrapolation (left) and Further Interpolation (Right). The blue data points indicate known data points, the orange data point is too be predicted. All data points are plotted by their solar elevation over the corresponding solar azimuth.

One aspect that shows why this assumption might be incomplete is visualized in Figure 23. The left plot shows a prediction data point (orange) close to multiple training data points (blue), whereby the right plot shows a prediction data point surrounded by multiple training data points of further distance. If distance were the most relevant factor in training prediction the left prediction point should be of better prediction accuracy. The left data point however is completely extrapolated from the known training data, while the right data point is interpolated. Therefore, it is likely that the right data point might be predicted just as well as the left one even if the distance between testing and training data is larger.

4.1.3. Quasi-Constant Alignment Behavior

The last assumption directly extends the second assumption, of heliostat behavior being changed by distance over different heliostat alignments. For sufficiently close distances the heliostat behavior is assumed to be quasi-constant. This assumption allows the entire heliostat behavior to be split into multiple regions of constant behavior. As a consequence one datasample per quasi-constant region sufficies to train a model to predict the entire heliostat behavior. This assumption is motivated by the heliostat's behavior being continuous without unexpected fluctuations.

4.2. Long- and Short-Term Heliostat Calibration

Within this thesis the phrasing short-term and long-term heliostat calibration is introduced. Thereby, short-term regards heliostat models that are optimized to predict alignments of similar behavior to those the model was trained upon, while long-term evaluation tries to cover all possible alignment behaviors. For high amounts of training data no distinction between both cases can be made. For small datasets short-term calibration can't rule out overfitting and thus might not accurately display a model's performance.

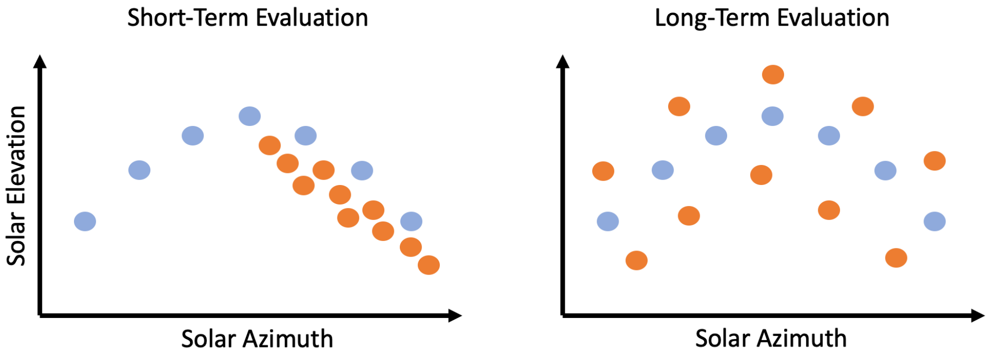

Figure 24: (Left) Short-Term Behavior Testing. (Right) Long-Term Behavior Testing. Blue data points indicate training data, orange data points the testing data. Short-Term evaluation focuses on a concentrated region while Long-Term evaluation regards testing data over the entire behavior range. All data points are plotted by their solar elevation over the corresponding solar azimuth.

All publications summarized in section 3 used time as a distance function between data points. Furthermore, by the above stated definition of short-term calibration and under the assumption of high dependancies between solar positions and heliostat behavior, only short-term calibrations were regarded.

This thesis instead focusses on long-term calibration. Therefore, within the next sections, a distance metric for heliostat alignment datasets based on solar positions is introduced. After deriving a metric suggestion from stochastics, a dataset splitting algorithm is suggested that allows for balanced datasets and makes overfitting more obvious. Eventually, a new approach for indicating alignment model performances is suggested based on the introduced metric and dataset splitting.

4.3. Alignment Behavior Distance Metric

Under the assumption introduced in section 4.1 a distance between heliostat alignment behaviors can be created by means of the euclidean distance over the correlating solar positions. Each solar position can be transformed into spherical coordinates if a constant heliostat concentrator position is assumed. Therefore, the vector between heliostat and solar position is computed and the solar position then stated as azimuth and elevation angle with arbitrary radius. The resulting distance function between two data points dp_1 and dp_2 is given in Equation 9.

$$dist(dp_1, dp_2) = \sqrt{(azim(dp_1) - azim(dp_2))^2 + (elev(dp_1) - elev(dp_2))^2} \quad (9)$$

4.4. Alignment Dataset Coverange and Balance Metric

A distance distribution can be computed for a dataset D_1 to a given set of solar positions D_2 via a Nearest Neighbor distance prototype (ref. section 2.11.1) by Equation 10. Such a set of solar positions could either be solar positions within a performance dataset created for model training or solar positions relevant to the heliostat's operation. This approach can further be expanded to k-Nearest Neighbors, relevancy weighted distances and more dimensional distances. Including more input parameters to the distance would require dimensional scaling and exceeds the scope of this thesis. The same applies for the introduction of relevancy weights.

$$nn\_dist(dp_i, D_2) = \min\left(dist(dp_i, dp_j)\right) \; \forall i \in D_1, j \in D_2 \quad (10)$$

Relevant solar positions can be defined as those solar positions correlated to a heliostat alignment's behavior region of interest (e.g. the upcoming month). Given a computed distribution of distances from relevant solar positions to a dataset, the dataset's balance and coverage can be quantified by the distribution's mean and variance. The mean indicates how equal all behaviors of interest are covert, while the variance shows the coverage's balance of the relevant behavior region. An ideally balanced dataset therefore would have a mean and variance of zero and thus completely cover all alignment behaviors. In reality, alignment behaviors are continous and thus can't be completely covered by a finite amount of data points. Therefore, the optimality objective for each alignment dataset is to minimize both the mean and variance to the greatest extent possible.

4.5. Dataset Splitting Algorithm

By means of the metric introduced in section 4.4, an ideal testing dataset's distance distribution would have a mean and variance of zero to completely cover all behavioral patterns. The same holds true for the validation and training dataset. Consequently, no ideal distribution can be achieved if dataset splitting is applied. Instead data splitting can be chosen to be as optimal as possible by reducing the dataset distributions' mean to cover all behavioral patterns and ensuring dataset balancing by reducing the distributions' variance.

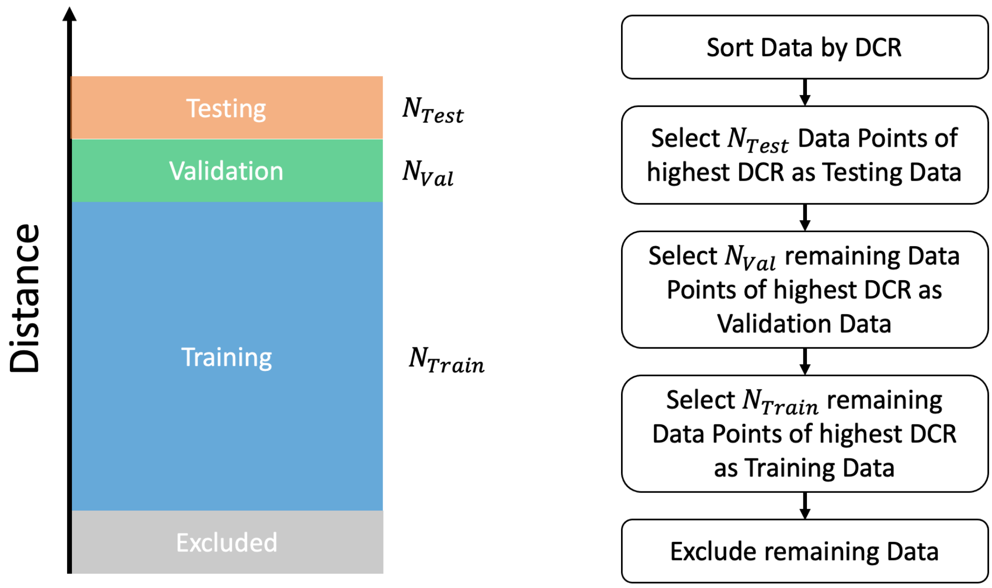

Figure 25: Data Splitting Algorithm. The right subplot summarizes the algorithm's processing steps. The left subplot indicates the resulting dataset split distribution over data point distance to any other data point. The distance arrow indicates increasing distances.

A heuristic solution to this optimality problem can be achieved by the algorithm shown in Figure 25. First all data points are sorted by their k-Nearest Neighbor distance to any other available data point. Then, the n_{test} data points of highest distance are selected as testing data points. This ensures a good coverage of all sampled behavior regions with testing data.

Thereafter, out of the remaining data points that are not selected as testing data, again the n_{val} data points of highest distance are chosen as validation data. n_{test} and n_{val} should be chosen as low as possible to leave enough training data but sufficiently numerous validation data to cover all relevant solar positions and thus behavirol patterns. The training dataset is constructed from the data that is not selected as testing or validation data. If not all remaining data is selected to construct the training dataset, data points of high distances are prioritized.

Splitting data by the above described algorithm ensures high distances between testing and any other data and thus makes the testing prediction performance a more reliable indicator for the model's generalization capabilities. Furthermore, testing and validation data tends to be equally distributed over the entire range of solar positions and thereby balances both datasets.

4.6. Operational Data Sampling Algorithm

The algorithm introduced in section 4.5 is designed for splitting existing datasets in accordance to the metric introduced in section 4.4. However, it's not applicable for sampling data points during ST operation, because it offers no decision rule, which data point outside the existing dataset should be sampled next. Furthermore, datasets for ST operation don't require testing data, as the model's architecture's capabilities are generally proven prior to its application. In this chapter, an algorithm for operational data sampling based on the metric described in subsection 4.4 is introduced.

The suggested algorithm is derived from the concept of quasi-constant alignment behavior, first described in section 4.1.3. Under the assumption of equal sizes for all behavior regions, all relevant heliostat orientations can be divided into evenly distributed segments. Following the made assumptions, one training data point plus one validation data point per behavior region should be sufficient to fit the model to all behavior regions. The algorithm thus starts by selecting an arbitrary data point from all possible data points as first training data point. Thereafter, the next training data point is selected to be as close as possible to the prior selected training data point but outside of the already covered estimated constant-behavior region. This process is repeated until the entire behavior region of interest is covered by training data points. The same steps are repeated for constructing the validation dataset. When choosing the training and validation data attention should still be paid to achieve sufficient data set seperation to reduce model overfitting on the given training data points.

5. AI Enhanced Dynamic Alignment Model

In this chapter a novel approach of enhancing heliostat alignment models with neural networks is introduced. The aim of this approach is to solve the problem of neural networks requiring large amounts of training data to learn a heliostat's behavior, by combining a neural network's ability to generalize patterns with the standard approach's low amount of required training data. A further focus lies on having the AI predict as little as possible and thus solely manipulate physically motivated deviations from an averaged optimum. To make this approach applicable to any heliostat of two or less actuators and potentially even more, the heliostat kinematic is regarded as a robotic manipulator and modelled according to the robotic industries standards.

Within the next subsections, the novel approach's conception is first discussed in context to existing solutions including a discussion of the approach's benefits over its competitors. Thereafter, an in-depth explanation of the new alignment model's inner structure is given.

5.1. Model Conception

The basic concept behind the approach introduced within this chapter is splitting a heliostat's alignment model into a static and dynamic part. The static part thus acts like the Rigid-Body Alignment Model approach, by modelling a heliostat's ideal kinematics amended by constant deviation factors. The dynamic part in contrast computes deviations to the ideal behavior based on the heliostat's current state (e.g. orientation) and dynamically adapts the deviation factors.

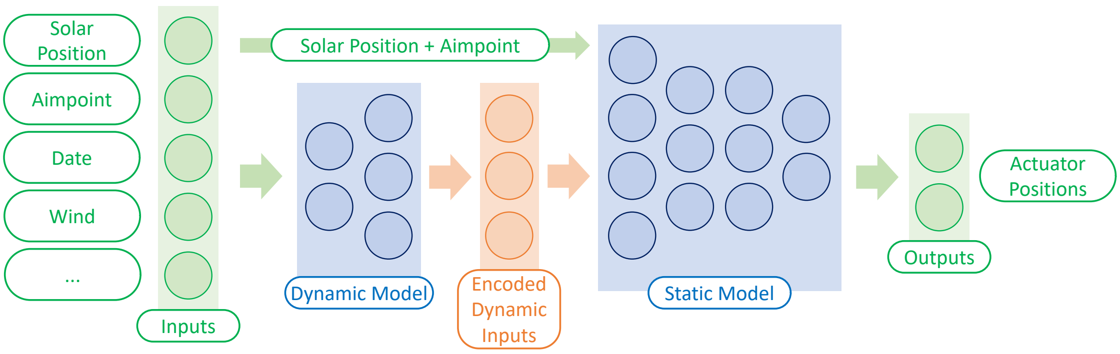

Figure 26: Concept of dividing static and dynamic heliostat behavior onto two neural networks (blue). All inputs are passed to the dynamic model that encodes the dynamic behavior into additional input values (orange) to the static model. The static model then uses solar position, aimpoint and the encoded dynamic behavior to predict correlated actuator positions.

In theory the above-described approach could be achieved by two subsequent neural networks, as visualized in Figure 26. The first network would act as a surrogate model. A surrogate model learns to approximate behavior for which the derivation of an analytical model is too complex. In this case the surrogate model would learn to translate given inputs into output parameters that encode dynamic effects onto the heliostat. Possible inputs are the solar position, aimpoint and environmental factors such as wind.

The second neural network could first be trained to predict only a heliostats ideal static behavior. By passing solar positions and aimpoints for multiple heliostats through the network and training it to predict associated actuator positions. Then, the static model could be retrained to also account for dynamic behavior by passing the first model's output parameters as encoded dynamic inputs to the second model. The encoded dynamic behavior thereby wouldn't need to be physically motivated parameters but would have to be interpretable by the second model.

The benefits of this approach are the same generality as a single Neural Network Alignment Model but for the static model being trainable on data for all heliostats of equal kinematic behavior, with only the dynamic model having to adapt to individual heliostats. Still, the static model would require to be trained to learn a behavior that can be modelled by the same techniques used to create the Rigid-Body Alignment Model. Hence, not only would valuable data be spent on training a neural network to learn mimicking the rigid-body approach, but also could this approach not be used to align a heliostat onto the calibration target prior to an initial training. Pretraining on simulated data could solve these issues, but would require creating a kinematic model for the heliostat alignment.

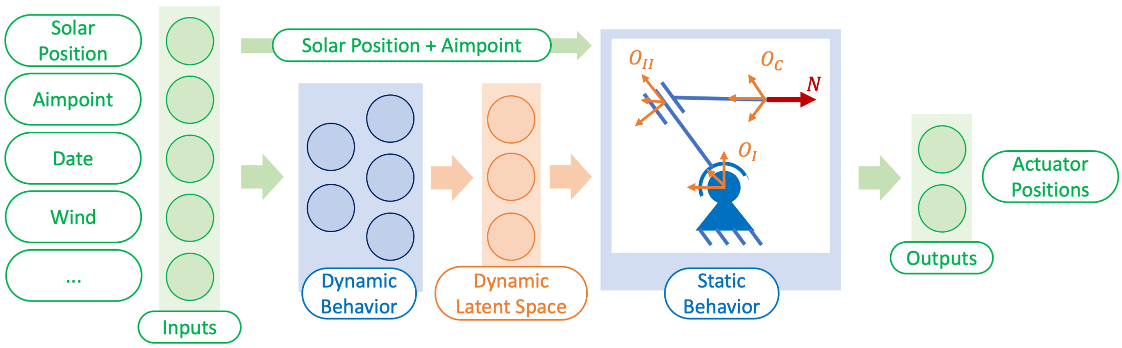

Figure 27: Dynamic and static heliostat behavior split onto two models (blue). The static behavior is modelled by analytically solving a rigid-body kinematic system, that computes actuator steps from solar position and aimpoint. The dynamic behavior is learned by a neural network. The neural networks encodes the dynamic behavior inputs into latent space variables (orange). These latent space variables are chosen to equal the kinematic model's parametrization and thus can adapt the rigid-body behavior to dynamic effects.

Thus, if a kinematic model must be created in any case, it could also directly be used to replace the static model entirely (ref. Figure 27). This approach is feasible because the dynamic model's output parameters can be trained to predict the kinematic model's geometry parameters. Instead of being constant, these geometry parameters thus are dynamically dependent on the heliostat's state. Using this approach, an initial precision boundary is guaranteed even without any prior training. Furthermore, dependent on the amount of available training data, different encoder models can be applied. The simplest encoder model is one layer of nodes that is not connected to any inputs and thus equals the Rigid-Body Alignment Model. Additionally, any kind of function (e.g. polynomials) that takes the heliostat state as input and predicts the geometry parameters can be applied. Therefore, future advances in heliostat physics modelling (e.g. structural bending under load) could replace parts of the encoder. The encoder is furtheron called dynamic model and the decoder static model.

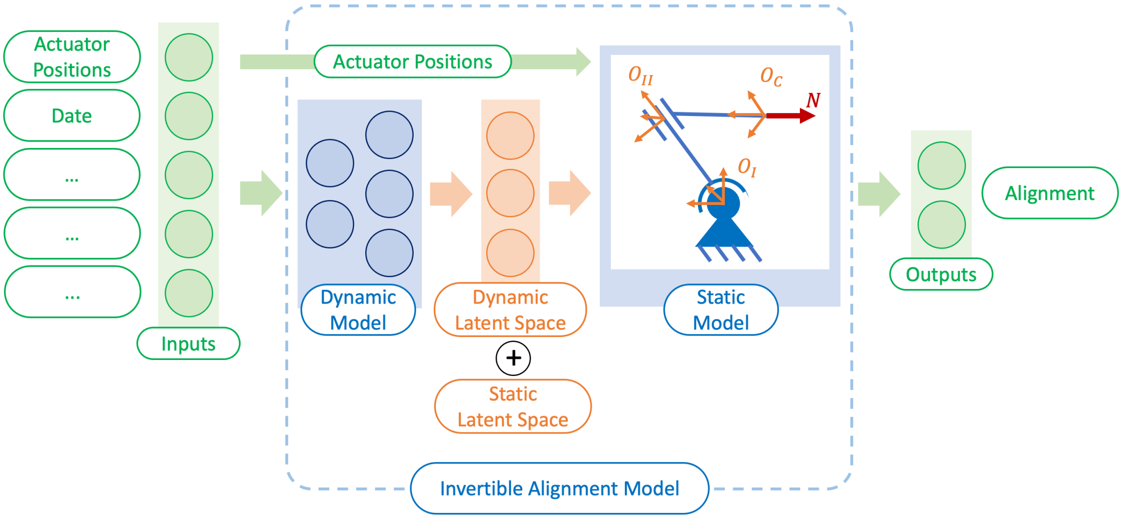

Figure 28: Dynamic and static heliostat behavior split onto two models (blue). The static behavior is modelled by analytically solving a rigid-body kinematic system, that the heliostat's alignment and thus mean concentrator normal and concentrator pivoting point from actuator positions. The dynamic behavior is learned by a neural network. The neural network encodes the dynamic behavior inputs into latent space variables (orange). These latent space variables are chosen to equal the kinematic model's parametrization and thus can adapt the rigid-body behavior to dynamic effects. To reduce the neural network's complexity, a pre-computed initial guess for the the latent space variables is added. Therefore, the neural network only needs to compute dynamic deviations from the initial guess. By choosing this modelling structure the entire alignment model remains invertible and thus can either compute an alignment from actuator steps or actuator steps from solar position and aimpoint.

Within this thesis an adaptation of the above-described approach is pursued and visualized in Figure 28. Instead of using source position and aimpoint as inputs and the actuator positions as outputs, actuator positions are chosen as inputs and the heliostat concentrator's alignment, pivot point and normal, as outputs. The reason for this adaptation is the kinematic model's general behavior and invertibility. In kinematic modelling, the forward direction, which is always directly solvable, computes a coordinate systems location and orientation based on all its parent coordinate systems locations and orientations, whereby each manipulatable coordinate systems is controlled by an actuated joint. Therefore, actuator positions to alignment is the kinematic model's forward direction and thus choosing actuator positions as inputs to the dynamic model aligns both models' predictions in the same direction. The forward direction hence is completely independent of a given source position and aimpoint. The inverse direction can mostly be computed in the same way as for the Rigid-Body Alignment Model as shown in Figure 29.

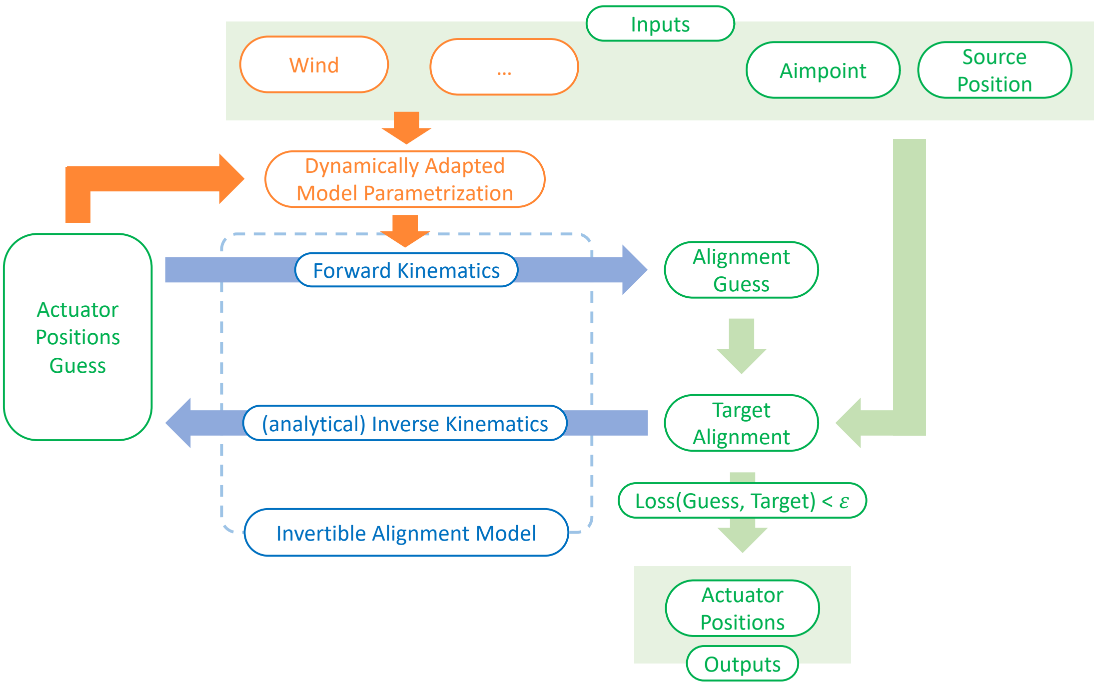

Figure 29: A heliostat alignment is computed by passing arbitrary actuator positions to the alignment model. Based on a given aimpoint and source position and the initial alignment's concentrator center, a target alignment is computed that follows the law of reflection. An updated actuator positions guess can be obtained via the model's inverse direction. A dynamic adaptation (orange) of the rigid-body kinematic model's parametrization allows the model to account for dynamic heliostat behavior. This adaptation is based on the actuator positions guess and additional dynamic inputs such as wind. This process is repeated by computing a new alignment from the updated acutator positions until the alingment guess is sufficiently close to the target alignment that results from the law of reflection.

The only adaptation to the inverse direction computation is the added dynamic model shown in orange and the splitting of input values into heliostat state inputs (e.g. date) and source position and aimpoints. While the latter are used to compute a target alignment, the state inputs influence the dynamic model's outputs. The dynamic model is part of the alignment approximation loop and thus predicts the geometry parameter deviations for each actuator position guess until the targeted and guessed alignment converge.

Another applied adaptation is the introduction of two-fold geometry parameters to the kinematic model that combined parametrize the model: One parameter holds the kinematic model's static rigid body behavior and one introduces dynamic deviations to the rigid body behavior and is controlled by the dynamic model.

Within this thesis the kinematic model is modelled as rigid body kinematics in the same way as robotic manipulators, where each joint forms the origin of a new coordinate system. Each coordinate system can be rotated and translated around all three axes but is part of the coordinate systems chain and thus has six degrees of freedom. Furthermore, manipulating a coordinate system higher up the chain affects all its child systems. Using this approach allows to compute a heliostat's alignment in global coordinates. For this thesis heliostats with two or less actuators are regarded, because manipulators of higher degree would require a numerical solver for computing the inverse direction, whereas simpler manipulators' inverse directions can be solved analytically. Therefore, each heliostat has two actuated coordinate systems at each joint plus an additional coordinate system at its concentrator's pivoting point. The actuators are modelled as described in section 2.8.1. This results in a total of 28 geometry parameters characterizing each heliostat.

5.2. Dynamic Model Output Scaling

As the alignments deviation is known to be within mrad magnitude the dynamic model's outputs are scaled accordingly. For rotational output parameters one output unit thus equals 10mrad. Translatory parameters are harder to scale. In this thesis spherical coordinates are chosen. The radius is scaled by a ratio of one output unit to 1m while the two angles are scaled at 1Rad per output unit. All percentual outputs are scaled by 1% per output unit. The reasoning behind the chosen scaling is to restrict most outputs expected values within a range of [-1,1]. Appendix A gives an overview over all parameters, their type and scaling.

5.3. Dynamic Model Neural Network Architecture

In general, the dynamic model is just a wrapper for multiple functions that predict a disturbance value based on given inputs. Within this thesis, a neural network is used as dynamic model. This has two benefits: The first benefit is that no knowledge about the disturbance patterns behavior and thus polynomial degree must be given, as the network will approximate the pattern's function by connecting its neurons to any function that fits the pattern best. Secondly, all disturbance parameters are decoded from an underlying encoding within the neural network and thus interconnect rather than being modelled individually.

One major design aspect of the dynamic model is its limited influence on the alignment model's behavior as it only predicts the deviations from the ideal assumption. This should as well be included within the chosen neural network architecture. Therefore, a Self-Normalizing Neural Network (SNN, ref. section 2.13.1) is chosen. A SNN is constructed to expect output values of zero with standard deviations of one and thus fit the output scaling described earlier in section 5.2. The SNN's number of inputs and outputs are fitted to the desired model configuration, while the number of hidden layers and neurons per layer is subject to hyper parameter optimization.

5.4. Dynamic Model Input Encoding

As summarized in section 2.12.2, it is good practice to normalize input parameters in machine learning applications. In the case of heliostat alignment disturbance models three different input parameter categories can be differentiated. These categories are "Heliostat Orientation", "Time" and "Environmental Conditions". Within the scope of this thesis only the first two categories are further discussed as no environmental data was available.

Due to the periodic behavior of time and date it holds two different types of information. The first type is its normalized duration Δt_n from a given date t_0, which can indicate the amount of wear on a system. The normalized duration can be derived by interpolating the current date t between an earliest date t_0 and latest date t_1 (ref. Equation 11). Then the normalized duration is encoded into a sine and cosine function.

$$\Delta t_n(t) = \frac{t - t_0}{t_1 - t_0} \tag{11}$$

The second type of information is the season, which can be shown as normalized month m_n that is constructed by dividing a data points month m by twelve. This input can indicate seasonal biases. This could be corrections to the solar algorithm or environmental impacts, whereby the latter should be used as individual inputs (e.g. wind or temperature) if available. Daily cycles are already encoded within the solar position and thus given through the heliostat's orientation.

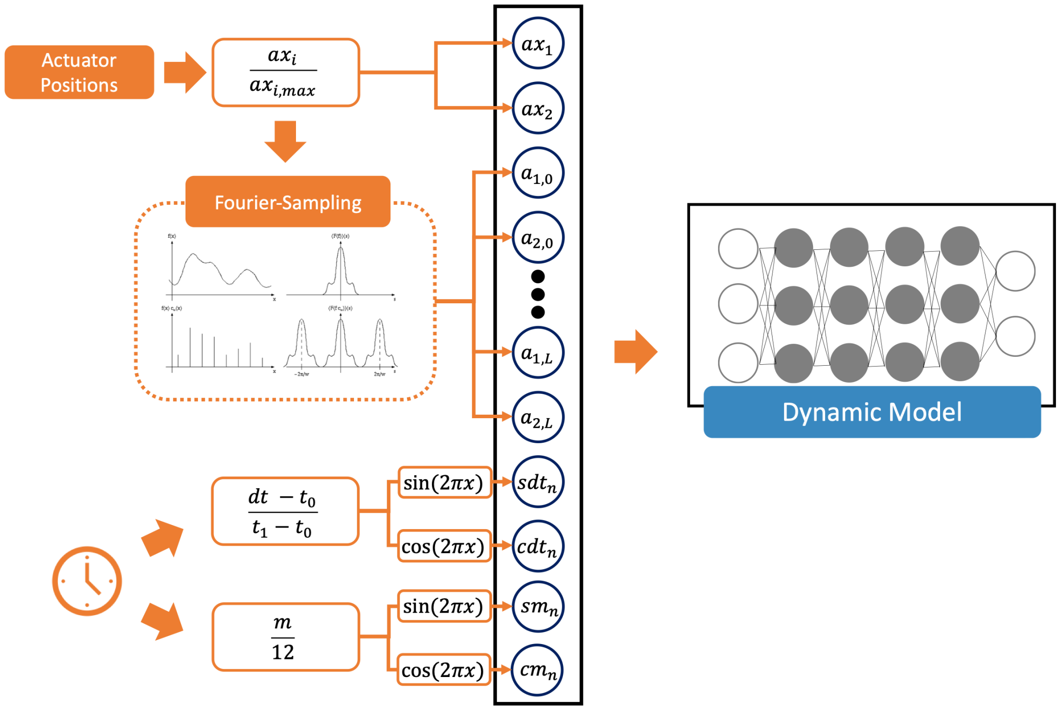

Figure 30: Dynamic Model Input Encoding. All inputs are first normalized and then encoded. For actuator positions Fourier Sampling is applied. All other inputs are encoded into their sine and cosine representations. The inputs are then combined into an input tensor that is passed to the dynamic model to predict dynamic parameter adaptations for the kinematic model.

The other input category of relevancy to this thesis is "Heliostat Orientation". The heliostat orientation can be extracted from a heliostat's actuator configuration. Therefore, a data point's actuator steps a_i are normalize to $a_{i,n}$ by dividing a_i by its maximum allowed value $a_{i,max}$ for each actuator i. After normalizing the actuator configuration, Pseudo-Fourier input encoding is performed according to section 2.12.2. Therefore, each actuator orientation is split into multiple frequency samples plus its original value increasing the number of inputs per actuator depending on the encoding degree L by $2L+2$. All time inputs are split into sine and cosine which is a simplified form of the positional encoding summarized in section 2.12.2. Figure 30 gives an overview of all encoded disturbance model inputs used in this thesis.

5.5. Dynamic Model Pre-Alignment

One major benefit of the approach suggested in this chapter is a technique further known as “Model Pre-Alignment”. By nature of the dynamic model’s design, it only computes deviations from the behavior that is expected by an alignment model’s static parametrization. Furthermore, different dynamic models can be applied to the same alignment model. Therefore, a two-step procedure, as visualized in Figure 31, can be performed that first computes static deviations that are then added to the alignment model’s static parametrization. This step equals the standard rigid-body optimization approach and can be achieved by using a a dynamic model with only one layer that is not connected to any inputs. The second step then uses a neural network as dynamic model to predict dynamic deviations to the pre-aligned alignment model parametrization. This reduces the AI’s impact and at the same time overcomes the rigid-body model’s restriction to averaged parameter optimization.

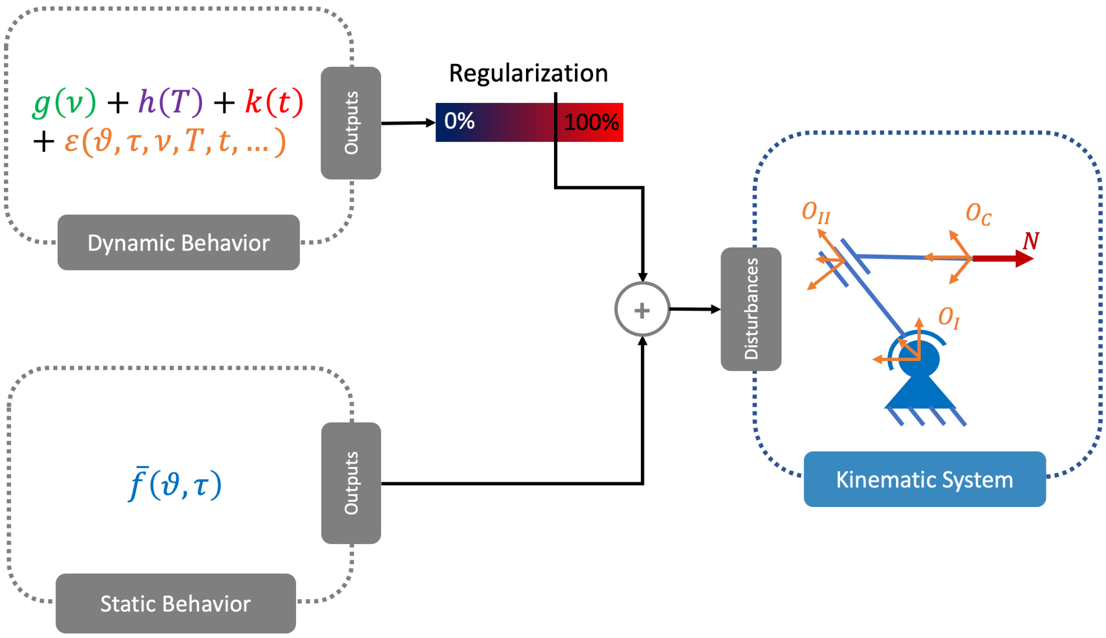

Figure 31: A heliostat’s behavior is split into a static and dynamic part. The dynamic part computes deviations based on the heliostat’s current state to the static assumption. Using this principle regularization can be applied to reduce the dynamic behavior’s impact on the adapted parametrization that is passed to the kinematic model.

Another major benefit to “Model Pre-Alignment" is the ability to regulate the AI’s impact factor. This can be achieved by applying weight decay regularization [comp. Chapter 2] to the dynamic model, especially neural networks. The regularization factor thus can either be adapted depending on the amount of available training data or swept over a pre-determined range. For high regularization factors the neural network’s complexity is penalized and therefore all model parameters converge against zero together with the models output values. Thus high factors result in small deviations from the optimized static parameters, whereas low regularizations factors allow unbounded network parameters and thus high deviations from the static parameters. In other words, regularization allows to controll the scope of solutions the dynamic model is allowed to find, with high regularization limiting the model to solutions close to the one found during pre-alignment. Furthermore, the performance of the AI combined with pre-alignment can be evaluated against the pre-alignment without AI and thus the chosen disturbance model is never worse than the current state of the art.

6. Datasets

All datasets used within this thesis were collected at Solar Tower Jülich (STJ) at 50.913525°N 6.387389 °E between Mai 2021 and October 2022. Aimpoints were derived via the Camera Target Method. Datasets for heliostats AJ.23, AM.35 and AM.43 were selected for analysis.

6.1. AJ.23

Heliostat AJ.23 is located 57m west and 66m north of the tower. Its earliest data point was sampled on June 2nd 2021 and its latest on October 28th 2022, resulting in 477 available data points, as visualized in Figure 32. In June and July 2021 a dense data distribution can be observed, which is a results from a specific measurement campaigns. However, no further information about the campaigns purpose or settings is available.

6.2. AM.35 and AM.43

Heliostat AM.35 is located 4m west and 80m north of the tower. Its earliest data point was sampled on Mai 5th 2021 and its latest on October 27th 2022, resulting in 191 available data points (ref. Figure 33).

Heliostat AM.43 is located 31m east and 80m north of the tower. Its earliest data point was sampled on August 11th 2021 and its latest on October 27th 2022, resulting in 198 available data points (ref. Figure 34).

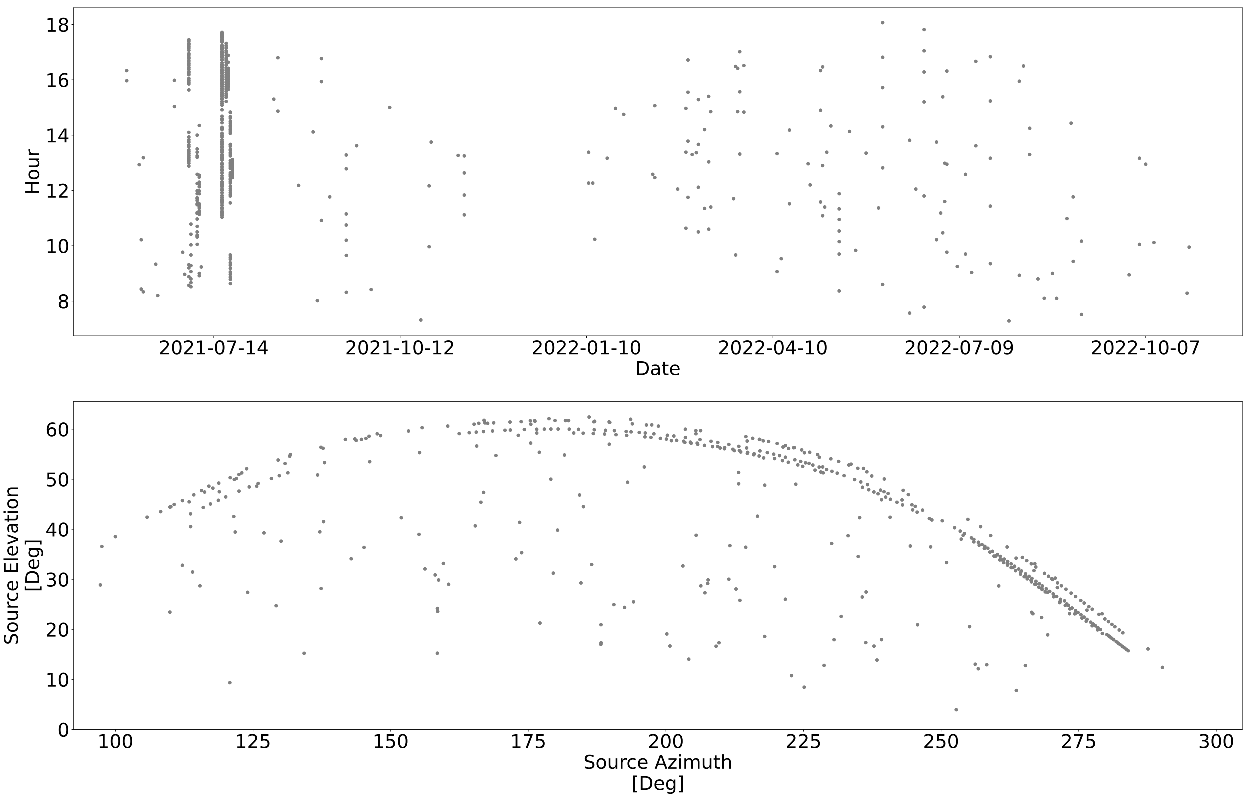

Figure 32: AJ.23 Data Points over Date and Solar Positions

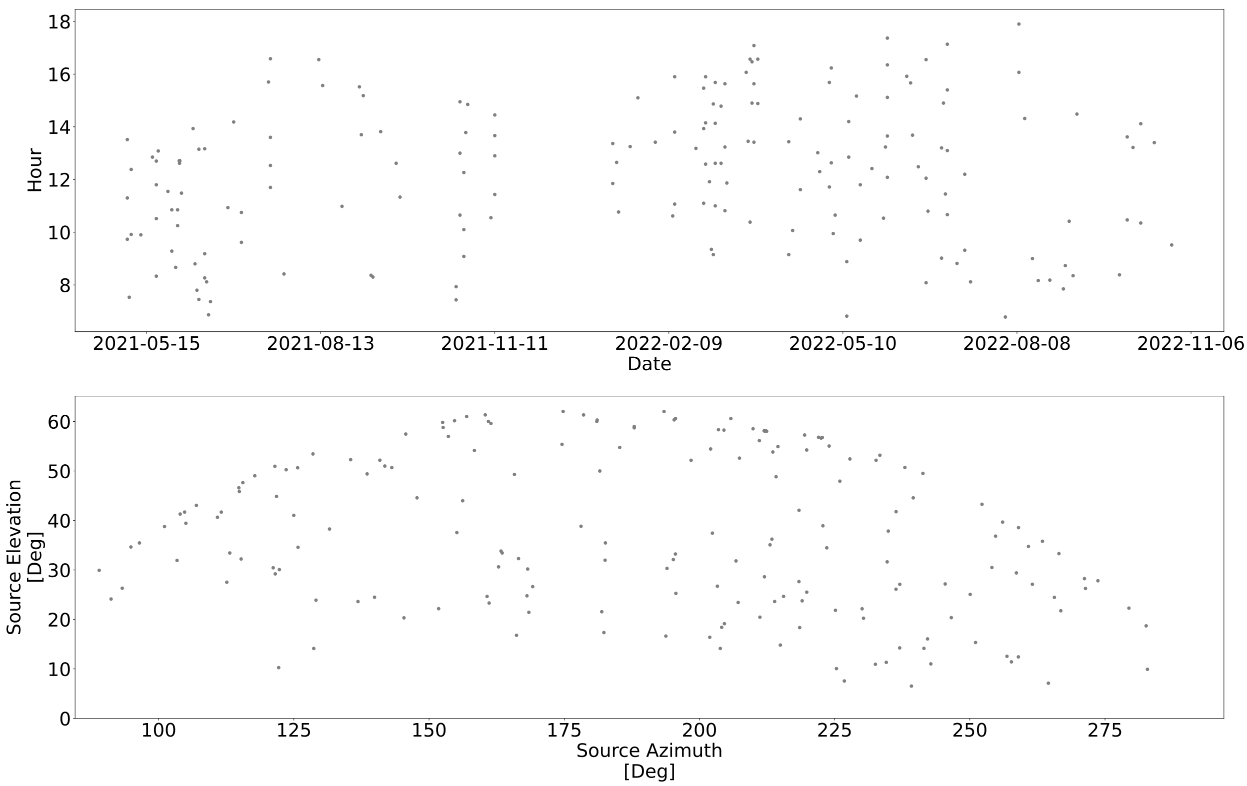

Figure 33: AM.35 Data Points over Date and Solar Positions

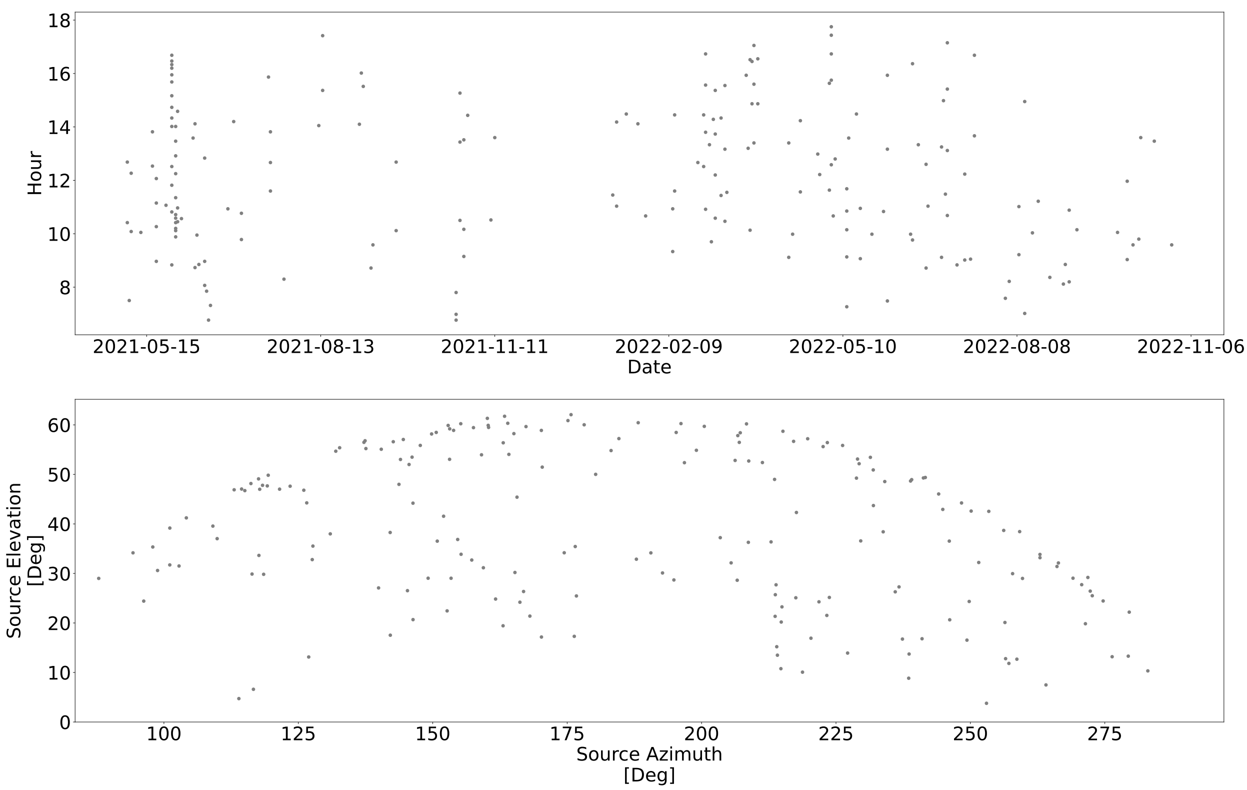

Figure 34: AM.43 Data Points over Date and Solar Positions

7. Experiment Results

This chapter aims to evaluate the performance of the kinematic model with neural network dynamic disturbances, introduced in section 5 in comparison to the rigid-body kinematic model. In contrast to the short-term evaluation approaches summarized in section 3 a long-term analysis as described in section 4 is pursued. Before assessing the performance of the model, the assumptions stated in section 4 regarding the significant impact of solar positions on model performance are validated. Thereafter, the model's performance and regularization capabilities are analyzed. Eventually, the algorithm for training heliostat alignment models on datasets of minimal size is tested.

7.1. Quasi-Constant Behavior Regions Evaluation

This chapter investigates the effects of optimizing a model onto one data point. Fitting a model to one data point makes this data point simultaneously training and validation data, while all other data points are testing data. This leads to the model being over-fitted to represent the given data point as good as possible. As a consequence, the model becomes an indicator for analyzing the heliostat's change in behavior over the entire dataset.

Figure 35 visualizes the model's prediction performances. The upper subplot shows the dataset distribution over date (x-axis) and hour (y-axis), whereas the center subplot visualizes the distribution plotted over solar azimuth (x-axis) and solar elevation (y-axis). The trained model's prediction accuracy on the test dataset is indicated by the plotted testing data point markers' size. The lower (=better) the accuracy the smaller the marker. Furthermore, all test data points are color coded to indicate their temporal position within the dataset. Blue indicates early data point and green the latest ones. The training data point is marked by an orange star. A tendency of prediction errors increasing over the data points' distance to the training data point can be observed in both the upper and center subplot. Especially, continuously low prediction errors are found within the training data point's vicinity in both plots. In the upper subplot this vicinity includes a timespan of approximately five hours over 1.5 months. It is important to notice that testing data points of similar hour to the training data point are more accurately predicted, even if multiple days lie between training and testing data, than data points of higher hour deviations. In the bottom subplot the training data point's vicinity can be characterized more linearly as both axes are of equal unit and scaling. Thus, the vicinity's radius R equals the distance $D(az, el)_{P_0}$ for the given training data point $P_0 = (az_0, el_0)$ as formulated in Equation 12.

$$R = D(az, el)_{P_0} = \sqrt{(az - az_0)^2 + (el - el_0)^2} \tag{12}$$

It can be observed, that those testing data points of furthest distance $D(az, el)_{P_0}$ to the training data point conincide with the highest prediction errors. This correlation is additionally highlighted by visualizing the testing data points' distance $D(az, el)_{P_0}$ to the training data point P_0 over the prediction accuracy as shown in the bottom subplot of Figure 35. The datapoints' color-coding again indicates their temporal order. Mixed clusters of early as well as late data points are found at similar distance-accuracy-combinations. One aspect to be noted is the mostly linear relation between accuracies and distances. The only exception to the linear behavior is a branched segment of data points with small distances between

0.5mrad and 1.0mrad that show slightly worse performances (7mrad to 12mrad) as would be expected from the otherwise linear relation.

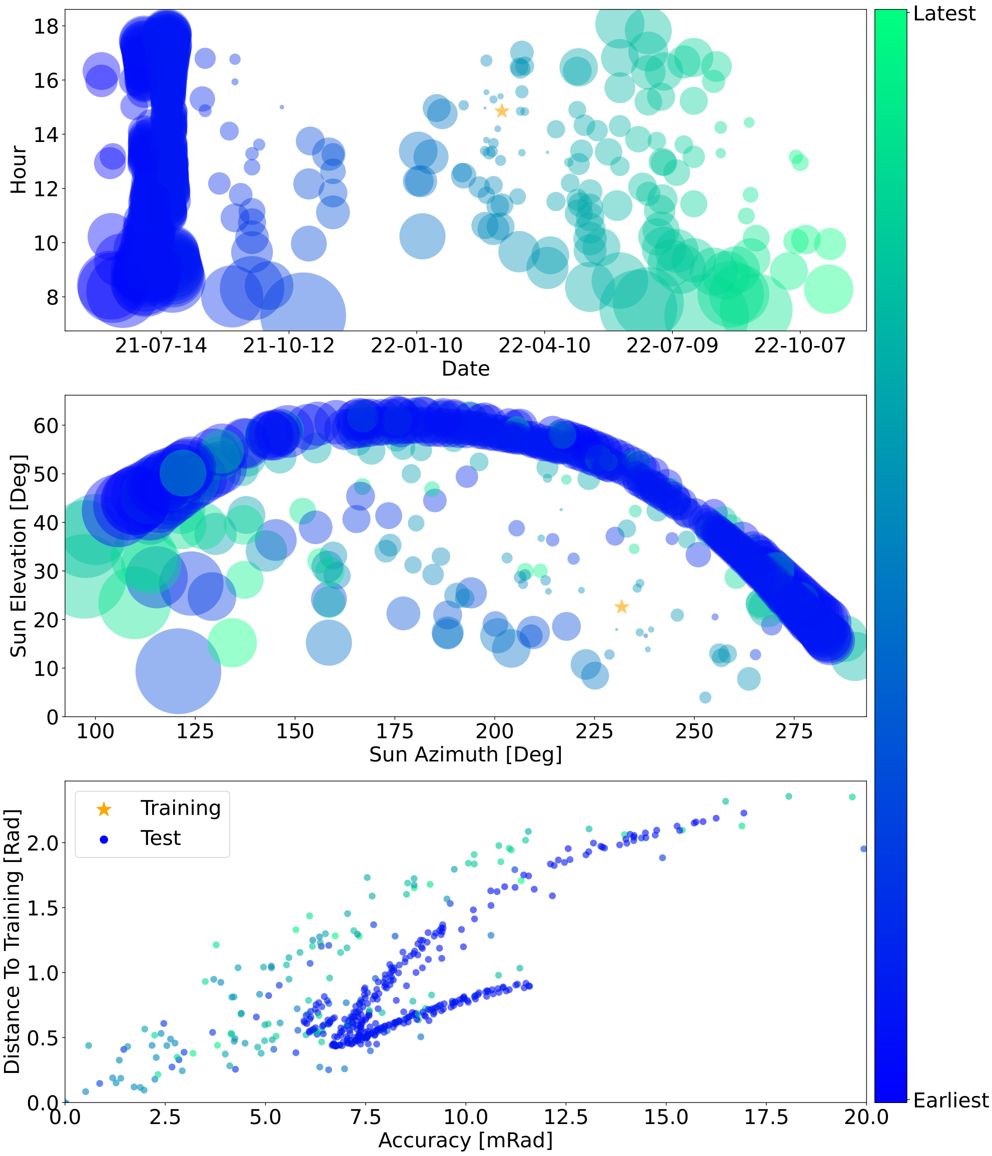

Figure 35: Rigid-Body Alignment Model prediction accuracies on the AJ.23 dataset after being trained on one data point (orange star). The prediction accuracy is indicated by the markers' sizes. The larger the marker the higher the error. All testing data points are color-coded by their temporal position within the dataset, with blue indicating early and green indicating late data points. The upper subplot shows the data points' creation of time distribution plotted by hour over date. The center subplot presents the data point's solar position distribution plotted by solar elevation over solar azimuth. The bottom subplot shows the data points' solar position distances to the training data point plotted over the achieved prediction accuracy.

7.2. Seasonal Effects

Section 4 implies that the ability of models to make accurate predictions primarily relies on the solar position. Models trained during a specific time of the year should therefore be able to predict heliostat alignments for the corresponding period in the following year. To prove this assumption, a rigid-body alignment model is trained on summer data and used to predict data for the following one and a half years. The used dataset is the AJ23 dataset introduced in section 6.1. Due to summer and winter having differing solar elevations throughout each day, seasonal prediction capabilities are thus expected, with good predictions in summer and lower performance in winter. Using the arguing of section 4, this can be explained by the constant elevation offset between training and testing data, that increases the average testing data point's distance to the training dataset.

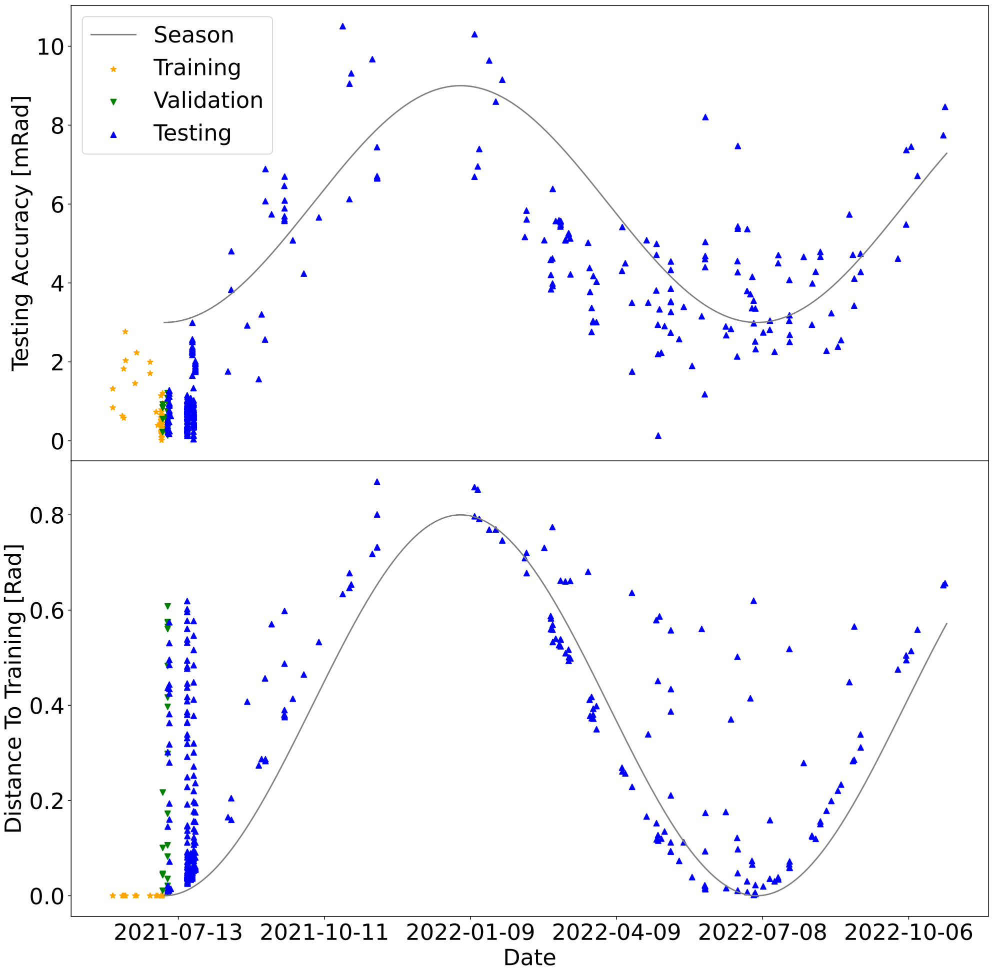

Figure 36: Rigid-Body Alignment Model performances (upper subplot) and distances to the training data (lower subplot) on the AJ.23 dataset plotted over time. The model is trained on the orange data points and validated on the green ones. Blue data points are testing data. A grey cosine of arbitray amplitude and yearwise frequency is added to visualize seasonal effects.

The seasonal effects are shown in Figure 36 and emphasized by plotting a cosine function with a frequency of one year and negative amplitude shifted to the first day of testing data.

The cosine functions mean and amplitude values were chosen to qualitatively resemble the seasonal effects, but have no other reasoning. From the upper plot of Figure 36 it can clearly be extracted that the model's prediction capabilities follow the cosine with good prediction results in summer and worse capabilities in winter. The lower plot of Figure 36 shows the corresponding distance distribution of datapoints to the training data, where again the seasonal effects do match the cosine.

7.3. Linear Dependency Experiment

To further analyze the correlation between time, solar position and (rigid-body) heliostat model prediction capabilities, performance evaluations on multiple datasets are performed. Each dataset thereby is constructed from 110 data points, split into 60 training, 20 validation and 30 testing data points. Different datasets are obtained by copying the last dataset's data points and then removing all data from the earliest currently covered day, before refilling the dataset with data from the following days. This process, resembles heliostat calibration during various seasons.

Two types of dataset splitting approaches are investigated. The first type is a chronologically ordered split. For this type, the earliest 60 data points are sampled as training data, followed by 20 data points for validation. The remaining (latest) 30 testing data points are selected as testing data. The second type splits the dataset spatially, using a k-Nearest Neighbor (NN) metric based splitting technique, as introduced in section 4.5.

Each dataset is then trained and its mean prediction accuracy on the testing dataset evaluated. To investigate the relation between dataset splitting approach and model performance the mean prediction accuracy is then divided by the dataset's mean distance between training and testing data. For chronological dataset splits, the temporal distance is chosen and for spatial dataset splits the k-NN distance. Eventually, all datasets are binned by mean prediction accuracy per distance and bin frequencies are counted.

The results of this experiment are plotted in Figure 37. The temporal bins (black) have highest frequencies close to mean prediction accuracies per distance values of zero, with decreasing frequencies for higher mean prediction accuracies per distance values. For this result 1-NN (cyan), 2-NN (green) and 3-NN (orange) dataset splits are evaluated. All three spatial dataset splits achieve a distribution with frequency means different from zero, but close to each other. Furthermore, the spatial dataset splits have lower variances, than the chronological split. h

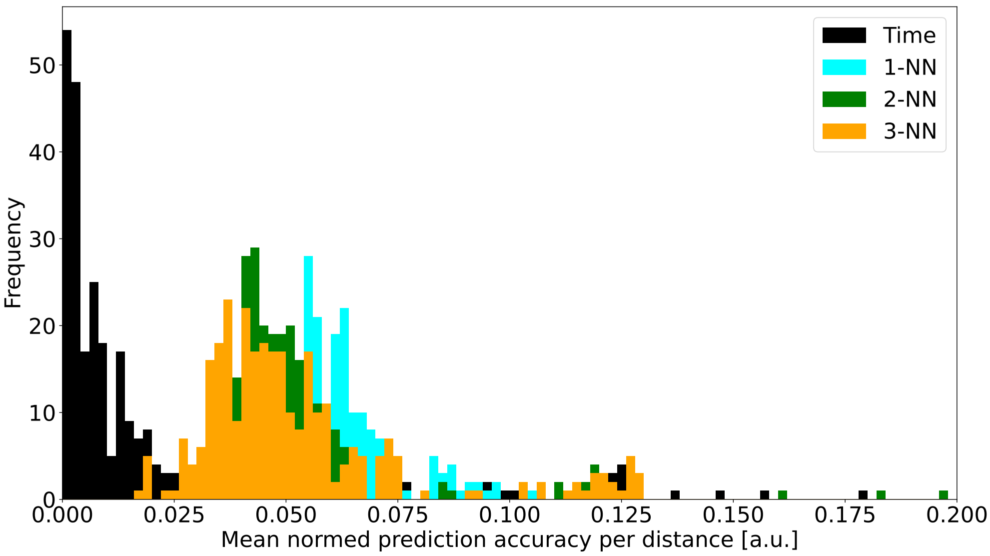

Figure 37: Frequency distributions of dataset splits on subsets of the AJ.23 dataset by time (black), 1-NN (cyan), 2-NN (green) and 3-NN (orange) plotted over prediction accuracies normed per distance.

Distribution means other than zero indicate a linear dependency between mean prediction errors and mean distances. From the non-zero distribution means for the spatial dataset splits, a dependency between spatial distance function and prediction accuracy can thus be deduced. No such dependency can in contrast be observed for the temporal dataset splits.

7.4. Alignment Model Evaluation

This chapter presents the evaluation results for the rigid-body, neural network enhanced and pre-aligned neural network enhanced alignment models introduced in section 5. First an overview of the general model and training parameter configuration is given. Then, the model performance depending on different training dataset sizes is analyzed for each of the datasets introduced in section 6. Furthermore, the influence of parameter regularization onto the pre-aligned neural network is discussed.

7.4.1. Model and Hyper-Parameter Configuration

For the experiments presented in this chapter, a fixed model and training configuration is chosen. First, some disturbance parameters are neglected due to parameter overlap and issues, where the optimizer pushed the actuator parameters into invalid parametrizations, that led to the law of cosine not being solvable. Therefore, out of all actuator parameters, only the actuator increments are regarded. Furthermore, all joint parameters are optimized, leaving a combined total number of 20 optimizable parameters.

A neural network configuration of 14 hidden layers with two neurons per layer was found to yield good results and thus is chosen. An input embedding of degree one is chosen for the actuator steps, resulting in six actuator inputs plus four additional time inputs to the neural network.

The training was found to yield best results for an initial learning rate of 0.1 with applied exponential learning rate decay of factor 0.997. Adam is chosen as optimizer. Cross-validation is applied and early-stopping based on the validation results is performed after 200 consecutive epochs without an improvement above 1% over the best prior epoch.

The above-mentioned hyper-parameters are chosen based on experience with the models' behaviors and thus have no claim of being optimal. Therefore, a future hyper-parameter sweep could potentially find better performing parametrizations but is not realized within the scope of this thesis due to time constraints.

7.4.2. Dataset Size Analysis

One common approach (e.g. [11]) of comparing alignment model prediction accuracies is to use the correlation between model abstraction capabilities and the amount of training data and thus indicating the performance over training dataset size. The concepts introduced in section 4 rather suggest a causal dependency between abstraction capability and distance between testing and training data. Therefore, within this section, different training dataset sizes are compared under the application of the dataset splitting algorithm suggested in section 4.5. 3-NN is selected as distance function. The 30 data points of highest distance are selected as testing data, the following 20 data points as validation data. Evaluated training dataset sizes are chosen pseudo logarithmically at 5, 10, 20, 40, 60, 100, 150 and 300 data points from the remaining data points until all available data points are selected or 300 training data points reached.

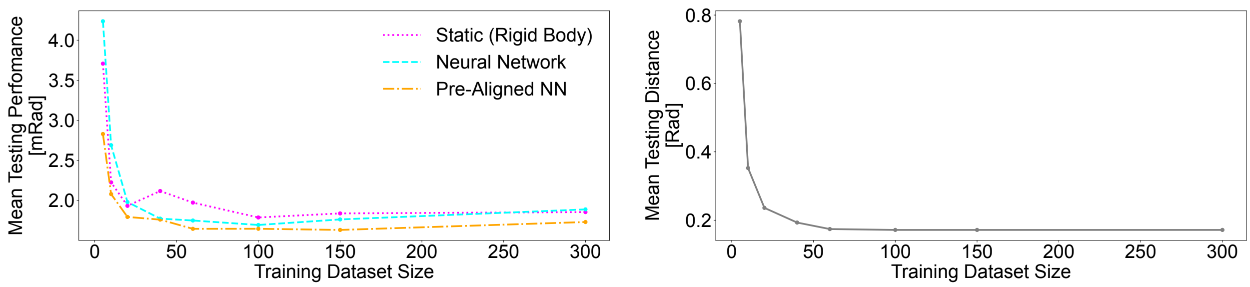

Figure 38: AJ23 Performance Comparison over Dataset Size

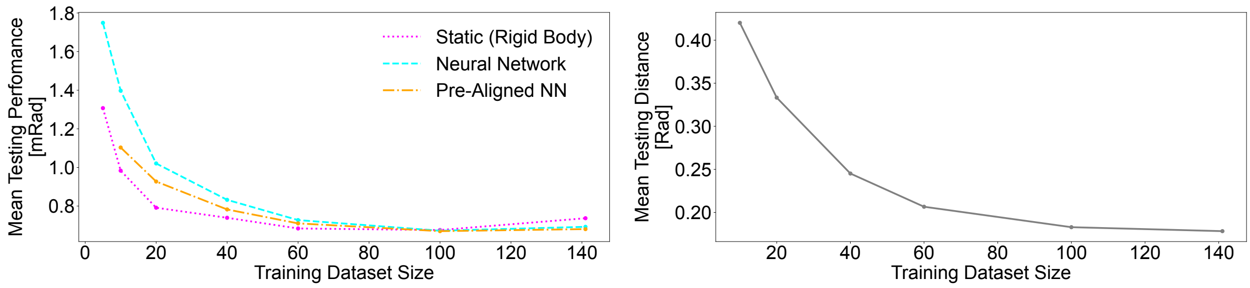

Figure 39: AM35 Performance Comparison over Dataset Size

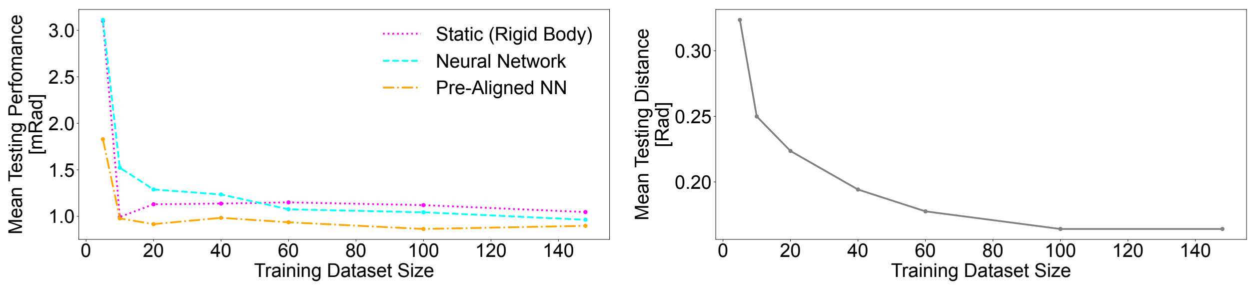

Figure 40: AM43 Performance Comparison over Dataset Size

Figure 38, Figure 39 and Figure 40 visualize the results for model trainings on the AJ23, AM35 and AM43 datasets. The left subplots indicate achieved mean testing prediction accuracies for models with rigid-body kinematic (magenta), neural network hybrid rigid-body kinematic (cyan) and pre-trained neural network hybrid rigid-body kinematic (orange) models, plotted over the amount of applied training data points. Major prediction accuracy improvements for additional added training data points can be observed for the first 20 data points for all models in all datasets. Thereafter, the rate of performance gains per added data point decreases. Starting from 100 data points a performance gain stagnation can be observed for each dataset. Furthermore, the best performing amount of training data points universally ain't the largest tested amount of training data. The rigid-body model outperforms the neural network hybrid model for small training dataset sizes below 20 data points for the AJ23 dataset, 40 data points for the AM43 dataset and 60 data points for the AM35 dataset. For larger amounts of training data, the neural network hybrid model continuously bests the rigid-body model. Except for the AM35 dataset, the pretrained hybrid model performes best for any given amount of data. On the AM35 dataset the pretrained hybrid model shows worse results than the rigid-body model for training dataset sizes below 60 data points, but beats it on larger training datasets. Compared to the neural network hybrid approach, the pretrained hybrid approach always showed better performances.

The right subplots in Figure 38, Figure 39 and Figure 40 show the mean metric distance between training and testing datasets over training dataset sizes. As expected by design of the dataset splitting algorithm, distances are rapidly reduced for small training dataset sizes and stagnate for larger training dataset sizes. All obtained distance curves show high resemblence to the testing performance plots in the left subplots. Notably are the training dataset size regions with mean testing distances between 0.2rad and 0.25rad. Within these regions the mean testing performance start to stagnate for all datasets, even though the distance might be further reduced for larger training datasets.

Figure 41, Figure 42, Figure 43, Figure 44 and Figure 45 show the datset distributions that result from applying the splitting algorithm for different training dataset sizes. The upper subplots show the data point time of creation distributions plotted by hour over date, the lower subplots show the data point solar position distributions plotted by solar elevation over solar azimuth. The data points marker sizes indicate the magniute of prediction errors linearly scaled by radius. For each data point a reference marker (grey circle) is introduced whose size equals a prediction error of 1mrad. As expected by the algorithm's design, testing (orange) and validation (green) data points are sampled from sparsely covered solar position regions. Therefore, almost all morning and evening data points are either testing or validation data. These "edge case" data points simultaneously appear to be the most difficult to predict for the alignment model, as their predictions have the highest errors. In Figure 41 the distribution for a training with 5 training data points (blue) is shown. Only a small solar position region is covered by these data points which is mirrored by good testing and validation prediction results in the region's vicinity and increasing prediction errors over distance to the training data points. This effect is reduced within the other Figures, where more training data points are applied.

Notably, is the high training data point density in higher solar elevations in Figure 45 and consequently good prediction accuracies within this region. Compared to Figure 43, especially the morning data predictions tend to be worse, even though five times more training data is applied.

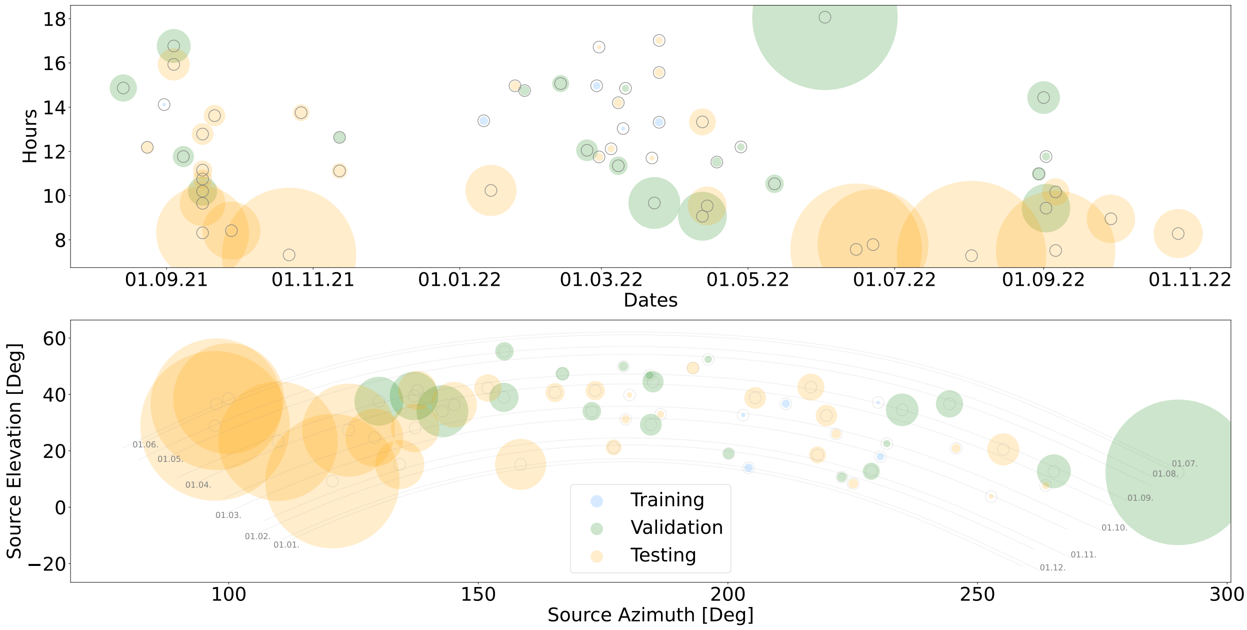

Figure 41: AJ.23 dataset split for 5 training data points.

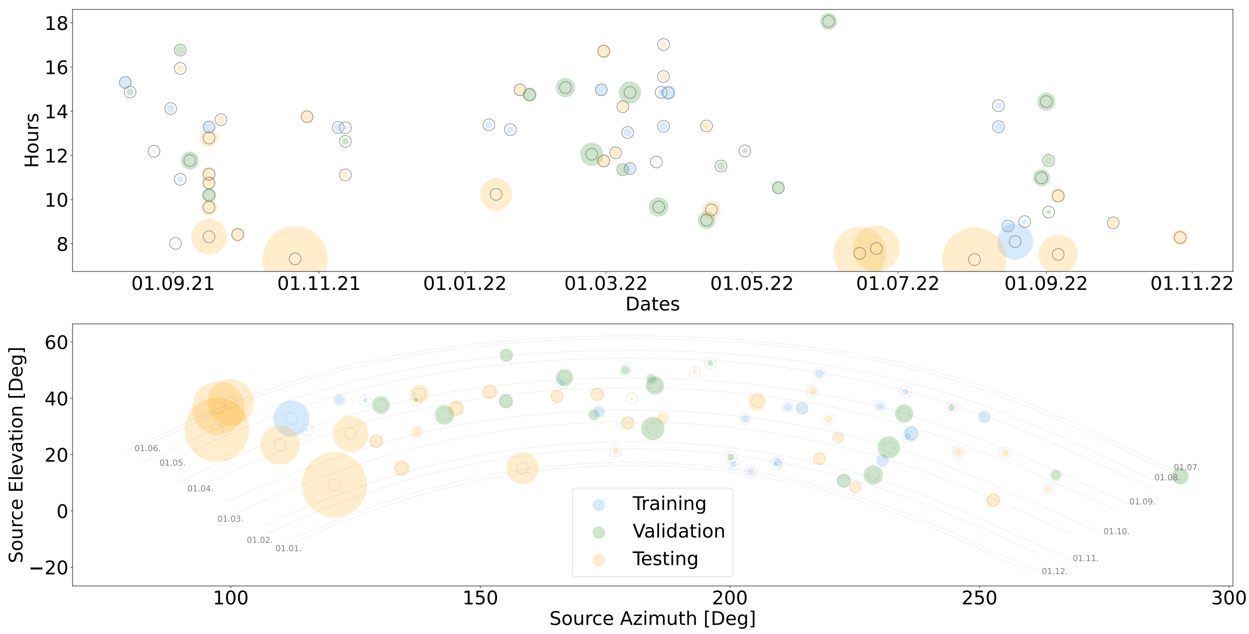

Figure 42: AJ.23 dataset split for 20 training data points.

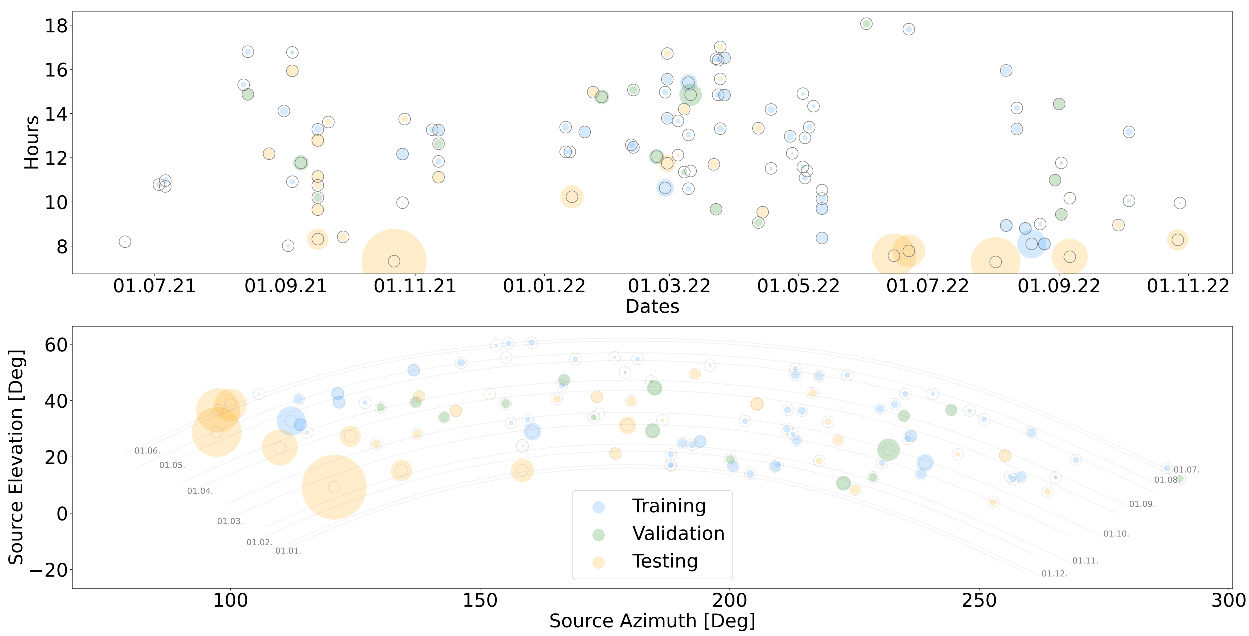

Figure 43: AJ.23 dataset split for 60 training data points.

Figure 44: AJ.23 dataset split for 100 training data points.

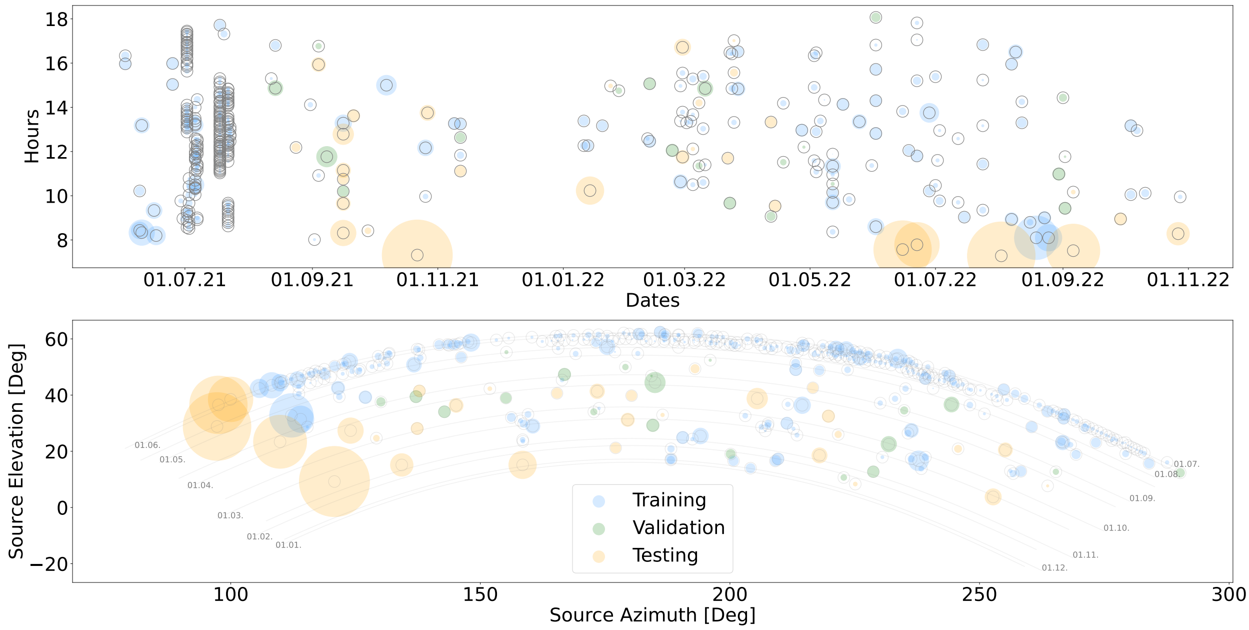

Figure 45: AJ.23 dataset split for 300 training data points.

7.4.3. Regularization Impact Analysis

As stated in section 5, a major benefit of the pre-aligned neural network approach is its regularization capability. Section 7.4.2 shows that AM.35 is the only dataset where the rigid-body model outperforms the pre-aligned neural network approach. In this case regularization (ref. section 5.5) can be applied to reduce the neural network's impact. In this section, the influence of regularization factors between 10^{-3} and 1 is analyzed. The same 60 training data point, 20 validation data point and 30 testing data point configuration for the AM.35

dataset as described in section 7.4.2, as well as the same model configuration as introduced in section 7.4.1 is selected. L1 regularization with varying weight decay factors is applied, and 20 trainings are performed and averaged per weight decay factor. The results are visualized in Figure 46. Mean prediction performances for the validation and test datasets are plotted over weight decay factors. Validation dataset results are shown by the magenta colored constant line for the rigid-body approach and the red line for the pre-aligned neural network approach. The testing dataset results for the rigid-body model are indicated by the blue constant line and the cyan colored line for the pre-aligned neural network model. While the pre-aligned neural network model's results converge against the rigid-body model's perforance for higher weight decay factors between 0.01 and 1, no such behavior can be observed for the validation results.

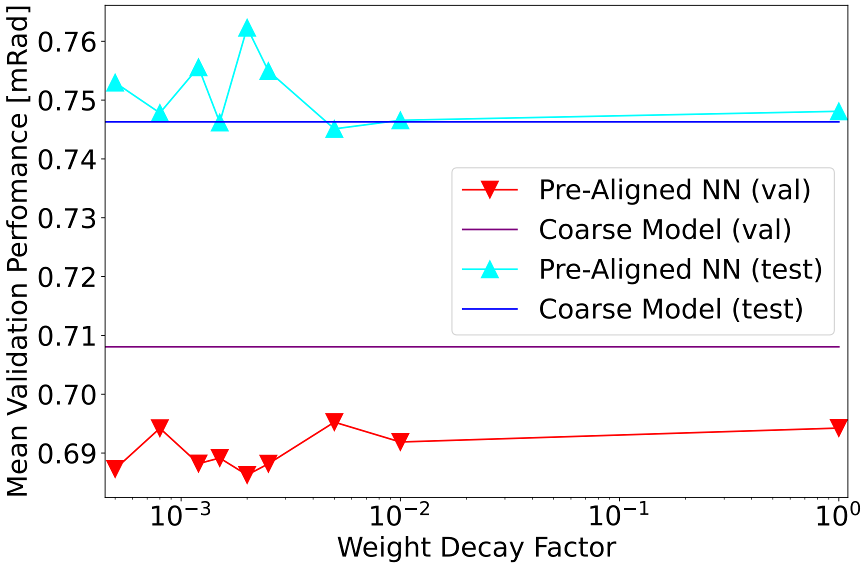

Figure 46: Pre-aligned model performances on a subset of the AM.35 dataset plotted over applied weight decay factors. The constant blue line indicates the rigid-body model's performance on the testing data points, the constant magenta colored line the rigid-body model's performance on the validation data points. The pre-aligned model's performances are plotted in cyan (testing data) and red (validation data).

7.5. Training Regions

As explained in section 7.2, seasonal prediction performance changes can occure due to a dataset imbalance. To mitigate this imbalance, training data should be distributed evenly over the entire solar azimuth-elevation range. Furthermore, the training dataset size analysis performed in section 7.4.2 indicated that most performance improvement were gained until an average distance below 0.25mrad was achieved between training and testing data. Together with the assumption of quasi-constant behavior regions introduced in section 4.1.3, the algorithm explained in section 4.6 can be applied. To prove that a small subset of data points is sufficient for training a model to predict an entire year, all available data points from the AJ.23 dataset for 2022 were chosen.

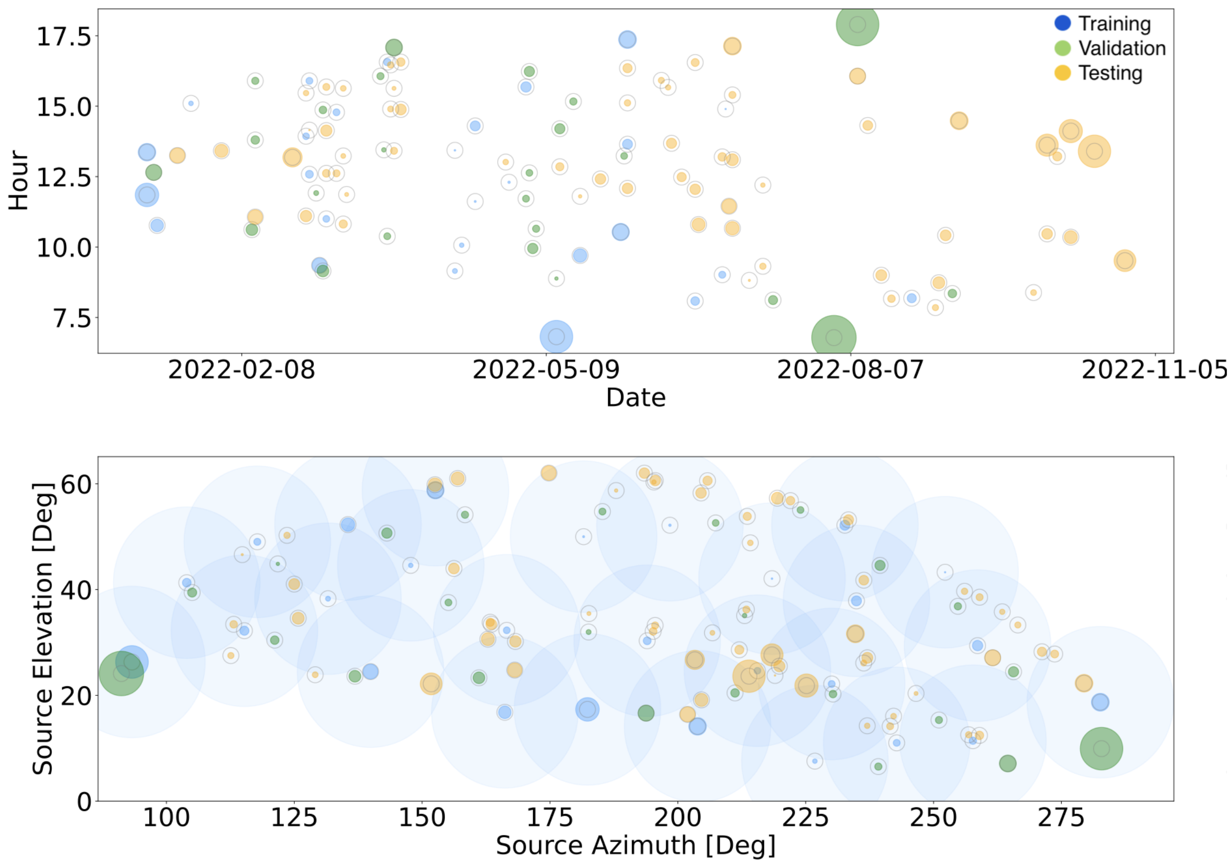

Figure 47: Achieved pre-aligned model predictions on the 2022 AJ.23 after being trained and validated on data points selected by the data point sampling based on regions. The upper subplot shows the data points' time of creation distribution plotted by hour over date. The lower subplot shows the data points' solar position distribution plotted by solar elevation over solar azimuth. The sampling regions are indicated as transparent blue areas. Prediction accuracies are indicated by the data points marker sizes. The larger the marker, the higher the prediction error. Reference markers (grey circles) are added to each data point indicating a reference prediction accuracy of 1mrad.

Figure 47 visualizes the resulting dataset split. All training data is shown in blue with surrounding blue regions of 0.25rad radius in the azimuth-elevation plot. As far as possible, given that the available dataset was not collected according to the suggested algorithm, the entire angle range is covered by the surrounding 27 training regions. The validation data, shown in green, is selected equally. All other data is selected as testing data, shown in orange. Two validation data points at the outer edges of the azimuth range appear show significant deviations from the average prediction performance and might be outliers. All testing data is predicted at an accuracy below 2mrad, with most testing data points having an accuracy below 1mrad.

8. Discussion

This chapter summarizes the findings of section 7 in the context of section 4 and section 5 with regards to section 3.

8.1. Temporal and Spatial Dataset Dependencies

Related work on the topic of heliostat calibration sofar suggests time as a major aspect to heliostat alignment model training (ref. section 3). The experiments conducted in section 7.1, section 7.2 and section 7.3 do not confirm this approach. Furthermore, section 7.2 even indicates that data points prediction accuracies for equal seasons remain stable for different years.

Instead, training a model on a single data point in section 7.1, indicates mostly linear dependencies between heliostat alignment model prediction performances and the solar position distance between the predicted data point and the training data point. Section 7.3 further shows a non-zero mean for the frequency distribution of mean predictions accuracies over spatial distances for multiple datasets, with training dataset sizes greater than one. However, this is no reliable indicator for a linear dependency between prediction performances for models trained on multiple data points and k-NN distances between training and testing data, due to the distribution's high variances. This coincides with the assumption of section 4.1.2 that the k-NN metric is a good estimator for model performances, even though it might be incomplete due to inter-data-point-effects such as inter- and extrapolation.

8.2. Training Dataset Size Reduction by Spatial Sampling

Contrary to the publications summarized in section 3, where the majority of testing data is distributed over a timespan of less than a month and thus the model is evaluated on only a subspace of a heliostat's entire behavior, this thesis focuses on evaluating a model's performance over the entire behavior range. Therefore, an algorithm for maximizing the distance between testing data points and training data points and therefore optimizing the testing data points coverage of the entire heliostat behavior range is introduced in section 4.5. Applying the algorithm in section 7.4.2 shows the desired effect of model performance gains per added data point stagnating, once the newly added data points don't reduce the distance between training and testing dataset. Furthermore, the resulting testing dataset is distributed over the entire range of solar positions and thus is a reliable evaluator for a heliostat's entire behavior.

8.3. Neural Network Hybrid Kinematic Model

The approach of first training a rigid-body heliostat alignment model on a dataset and computing dynamic deviations to its pretrained parametrization that is introduced in section 5, mostly outperforms the rigid-body approach for the datasets tested in section 7.4.2. It thus is an improvement over the state of the art approach without increasing the amount of required data points. Best model performances are observed for dataset sizes below 100 training and 20 validation data points. *Pargmann et al. (2021)* [11] in comparison found a pretrained neural network performing best on a training dataset of 300 data points. It has to be noted, that, whether this is a result of the new modelling approach or the dataset splitting algorithm remains to be tested at this point. Increasing dataset sizes above 100 data points leads to decreasing performances, when tested on the datasets of section 7.4.2. This appears to contradict the dataset metric introduced in section 4.4 as the model performance decreases while the distance remains equal. When analyzing the resulting dataset splits, however, an inbalanced training data distribution can be observed with a majority of data points being sampled from summer data, that results in lower elevation morning data being under represented within the training data. This is an indicator for overfitting on high elevation data.

Model regularization performed in section 7.4.3 generally showed the expected results of the pretrained neural network hybrid model converging to the same prediction performance as the rigid-body model. Still, a complete convergence could not be achieved. When regarding the model's performance on the validation dataset, the prealigned approach continuously outperforms the rigid-body model, which would not be the case for complete weight decay regularization. This results from the transient period at the start of training process, where the model's parameters deviate from zero due to the gradient descent fluctuations. These fluctuations lead to model parametrizations that outperform the rigid-body model on the validation dataset but are small enough to not impact the prealigned model's performance on the testing dataset.

8.4. Quasi-Constant Behavior Regions

Section 7.1 validates the assumption of section 4.1.3 by proving that the heliostat alignment behavior remains quasi-constant for small enough devitations over spatial distances. Section 7.4.2 indicates the model performance gain per added data point stagnating for distance decreasements within 0.25mrad to 0.2mrad. As a consequence, the algorithm suggested in section 4.6 is validated for an assumed quasi-constant region size of 0.25mrad in section 7.5.

9. Conclusion

The findings of this thesis affirm the quantifiability of heliostat alignment data point values to the model's performance capabilities and demonstrate the feasibility of modelling dynamic heliostat behavior with limited data. Contrary to the approach of most recent publications, the factor of time is found to be not directly correlated with a data point's value to model prediction accuracy. Instead a k-NN metric over the distances between a dataset's angular solar positions (azimuth and elevation) is introduced. It's shown that model prediction accuracies on a test dataset improve when the metric-distance between training and test dataset is reduced. Opposite to the common assumption of the model training being improved by adding more data points, prediction accuracy improvements stagnate when no further metric-distance reduction is achieved and can even deteriorate if a dataset bias is introduced. Still, a strict correlation between metric distance and model prediction performance could not be verified, as a metric distances of 0.2rad for example resulted in a prediction accuracy range between 0.7mrad and 1.7mrad for datasets of different heliostats. However, this is expcected as other criteria, such as interpolation and extrapolation effects influence the model's performance, which are not regarded within the suggested metric.

Until now, heliostat alignment research lacked the means for sufficiently evaluating a heliostat alignment model's performance. This leads to the testing data behaving similar to the training data, which can induce model overfitting and the publication of overly optimistic results. By means of the newly introduced metric, the applied dataset's prediction difficulty for an alignment model can now be quantified and more accurate estimations about the model's performance published. Moreover, if the distances between training and test data are disclosed, it now becomes possible to compare published results with one another. By selecting dataset splits that maximize the separation between training and test data, conservative estimations regarding the model's performance can be derived. For further disclosure of model performances a dataset performance plot is suggested, that combines a visualization of the temporal and spatial distribution of the applied dataset split, as well as the model performance for single data points.

The concept of quasi-constant heliostat behavior regions is introduced and validated. Based on the idea that one data point is sufficient per quasi-constant region, an algorithm for minimizing the required amount of training data to cover a heliostat's entire long-term behavior is derived. The found results demonstrate that an alignment model for heliostats can be trained to accurately predict the entire range of behavior using fewer than 30 training data points. For alignment models, where model overfitting is of concern, the same principles can be applied to construct the validation dataset. This algorithm can further be applied to heuristically prioritize heliostat selection during calibration.

A new approach of exchanging a heliostat's kinematic model's static parametrization by a dynamic disturbance model by means of a neural network is introduced. This approach can not only account for dynamic factors within the heliostat's kinematic, e.g. structural bending, but also extends the feasible input parameter space to include environmental impacts. In this thesis time was applied as environmental impact, but other factors such as wind could be applied with little adaptation of the model's architecture. Overall, the application of dynamic disturbance models outperformed the common kinematic alignment model approach. Using pre-alignment, where a model is first optimized to adapt its constant parameters to reduce the total prediction error as far as possible and then add dynamic disturbances to the static optimum, performed best. The combination of a state-of-the-art kinematic model with neural network dynamic disturbances further overcomes the issue of large training datasets that other neural network alignment approaches require.

Both the distance metric and dataset splitting algorithms as well as the pre-aligned kinematic model with neural network disturbances are integrated into the differentiable ray-tracing environment, developed by *Pargmann et al. (2023)* [12]. An adapted version of the heliostat calibration priorization algorithm is integrated into HeliOS [14] and already applied at the solar tower in Jülich. The scientific findings derived from this thesis are further summarized in two associated publications [41] [42].

10. Outlook

The introduced principles within this thesis to the approaches of heliostat alignment modelling enable further research that couldn't be investigated due to timely constraints. Outlooks for possible further investigations where collected throughout the conception of this thesis and are summarized in the following sections.

10.1. Hyper-Parameter Tuning

As model optimality was not the focus of this thesis, only an incomplete hyper-parameter sweep was fullfilled, leaving space for further hyper-parameter tuning. Especially, the influence of different neural network sizes, batch sizes, input-encoding degrees or global optimization could yield further performance improvements. Furthermore, all model parameters can be activated or deactivated for training. For small amounts of training data reducing the number of trainable parameters could improve the model's short-term prediction capabilities by focussing it to overfit on the given data.

10.2. Dynamic Model Architecture

Besides its self-normalizing capabilities the chosen neural network architecture for the dynamic model is not especially optimized. More sophisticated network architectures, such as modern foundation networks, were found to excel in multiple disciplines and might outperform the suggested approach. Therefore, an investigation into the application of modern neural network architectures is highly recommended.

10.3. Environmental Impacts and Structural Bending Analysis

One major benefit of the introduced approach is its capability to take environmental impacts such as wind and encode them into kinematic impacts on the heliostats structure. The datasets available to this thesis lacked environmental data and thus no further analysis on their impact was pursued. Possible applications to be investigated are a reduction of alignment deviations due to wind and temperature changes, clouds and humitidy and other environmental impacts. Furthermore, if given enough data, a trained dynamic model's encoded outputs could be used to analyze the impacts of gravity or wind on a heliostat's structural bending. Findings of such an analysis could be applied to improve future heliostat designs or derive and verify physical relationsships.

10.4. Heliostat Pre-Calibration and Cost Reduction

The suggested alignment modelling has complete controll over a heliostat's static and dynamic kinematics. It furthermore was shown that small amounts of data, sampled at the right heliostat orientations, can significantly improve a heliostat's alignment accuracy. Given an alignment sampling system that allows for high initial alignment deviations, the elaborated and thus costly pre-calibration step might be reduced to one initial sampling point. All further model optimizations could then be performed using standard alignment sampling systems by chosing data points of close solar position distance to the first sampled data point.

Another benefit that arises from the introduced modelling-approach's completeness is that impacts such as behavioral shifts over time due to wear that can't be mitigated by current systems can now be accounted for. Future heliostat designs might thus be constructed less steady and thus more cost-efficiently leaving additional disturbances to be reduced by the dynamic alignment model.

10.5. Alignment Dataset Balancing

While this thesis balanced alignment datasets by selecting data points using the solar position metric introduced in subsection 4.3, more sophisticated balancing techniques could be applied in future. [34] demonstrated a high correlation between a dataset's effective pattern density distribution and a model's prediction performance. To counteract the impact of majority patterns label distribution smoothing (LDS) is suggested, where the inverse pattern distribution density is used to re-weight the loss term. Another re-balancing technique suggested by [34] is feature distribution smoothing (FDS) where feature similarities for each pattern are analyzed and inserted into a feature calibration layer after the model's final feature map. Applying these approaches to heliostat alignment training could yield to further performance improvements.

10.6. Raytracing for Alignment Improvements

All presented results within this thesis applied the standard flux density center calibration technique, where a heliostat's reflection is reduced to a single aimpoint at the reflection's intensity center. This approach is a good approximation to calibrate a heliostat's behavior but discards a lot of information about the reflection's overall shape. [12] developed an environment were the heliostat alignment can be calibrated directly onto the reflective behavior. This thesis' findings are integrated into the raytracing-environment but no raytracing enhanced calibration was performed. Therefore, it is highly recommended to evaluate the benefits of raytraced-calibration as a next step.

11. Acronyms

BCS Beam Characterization System

NN Nearest Neighbor

AI Artificial Intelligence

DLR German Aerospace Center

CSP Concentrating Solar Power

MSE Mean Squared Error also called L2

CSP-OS Concentrated Solar Power Operating System

CSP Concentrated Solar Power

ST Solar Tower

DIR Deep Imbalanced Regression

12. References

[1] Ritchie, Hannah ; Roser, Max ; Rosado, Pablo: Energy. In: *Our World in Data* (2022), Oktober. `https://ourworldindata.org/energy-mix`

[2] Crellin, Forrest ; Aloisi, Silvia ; Chestney, Nina ; Chestney, Nina: Analysis: France braces for uncertain winter as nuclear power shortage looms. In: *Reuters* (2022), August. `https://www.reuters.com/world/france-braces-uncertain-winter-nuclear-power-shortage-looms-2022-08-30/`

[3] Reuters: Explainer: The power crunch in China's Sichuan and why it matters. In: *Reuters* (2022), August. `https://www.reuters.com/world/china/power-crunch-chinas-sichuan-why-it-matters-2022-08-26/`

[4] tagesschau.de: *Energiepreise ziehen hierzulande 110 Milliarden Euro ab.* `https://www.tagesschau.de/wirtschaft/finanzen/energiepreise-realeinkommen-wirtschaftsleistung-101.html`

[5] Bhutada, Govind: *The Raw Material Needs of Energy Technologies.* `https://elements.visualcapitalist.com/the-raw-material-needs-of-energy-technologies/`. Version: Juni 2021

[6] *Mineral requirements for clean energy transitions – The Role of Critical Minerals in Clean Energy Transitions – Analysis.* `https://www.iea.org/reports/the-role-of-critical-minerals-in-clean-energy-transitions/mineral-requirements-for-clean-energy-transitions`

[7] Wang, Zhifeng: *Design of solar thermal power plants.* 1st edition. Cambridge, CA : Elsevier, 2019. – ISBN 978–0–12–815613–1

[8] Sattler, Johannes C. ; Röger, Marc ; Schwarzbözl, Peter ; Buck, Reiner ; Macke, Ansgar ; Raeder, Christian ; Göttsche, Joachim: Review of heliostat calibration and tracking control methods. In: *Solar Energy* 207 (2020), September, 110–132. `http://dx.doi.org/10.1016/j.solener.2020.06.030`. – DOI 10.1016/j.solener.2020.06.030. – ISSN 0038092X

[9] Sarr, Marie P. ; Thiam, Ababacar ; Dieng, Biram: ANFIS and ANN models to predict heliostat tracking errors. In: *Heliyon* 9 (2023), Januar, Nr. 1, e12804. `http://dx.doi.org/10.1016/j.heliyon.2023.e12804`. – DOI 10.1016/j.heliyon.2023.e12804. – ISSN 2405–8440

[10] Monreal, A. ; Burisch, M. ; Sanchez, M. ; Pérez, D. ; Villasante, C. ; Olabarrieta, E. ; Olasolo, D. ; Olarra, A.: EASY: An Innovative Design for Cost Effective Heliostats/Solar Fields. In: *Energy Procedia* 49 (2014), 174–183. `http://dx.doi.org/10.1016/j.egypro.2014.03.019`. – DOI 10.1016/j.egypro.2014.03.019. – ISSN 18766102

[11] Pargmann, Max ; Maldonado Quinto, Daniel ; Schwarzbözl, Peter ; Pitz-Paal, Robert: High accuracy data-driven heliostat calibration and state prediction with pretrained deep neural networks. In: *Solar Energy* 218 (2021), April, 48–56. `http://`

dx.doi.org/10.1016/j.solener.2021.01.046. – DOI 10.1016/j.solener.2021.01.046. – ISSN 0038–092X

[12] Pargmann, Max ; Ebert, Jan ; Quinto, Daniel M. ; Pitz-Paal, Robert ; Kesselheim, Stefan: In-Situ Solar Tower Power Plant Optimization byDifferentiable Ray Tracing / In Review. Version: Februar 2023. http://dx.doi.org/10.21203/rs.3.rs-2554998/v1. 2023. – preprint

[13] Kuhn, Thomas: Digitaler Zwilling. In: *Informatik-Spektrum* 40 (2017), Oktober, Nr. 5, 440–444. http://dx.doi.org/10.1007/s00287-017-1061-2. – DOI 10.1007/s00287–017–1061–2. – ISSN 0170–6012, 1432–122X

[14] Gross, Fabian ; Geiger, Mark ; Buck, Reiner: A universal heliostat control system. Abu Dhabi, United Arab Emirates, 2017, 030022

[15] Kribus, Abraham ; Vishnevetsky, Irina ; Yogev, Amnon ; Rubinov, Tatiana: Closed loop control of heliostats. In: *Energy* 29 (2004), April, S. 905–913. http://dx.doi.org/10.1016/S0360-5442(03)00195-6. – DOI 10.1016/S0360–5442(03)00195–6

[16] Ulmer, Steffen ; März, Tobias ; Prahl, Christoph ; Reinalter, Wolfgang ; Belhomme, Boris: Automated high resolution measurement of heliostat slope errors. In: *Solar Energy* 85 (2011), April, Nr. 4, 681–687. http://dx.doi.org/10.1016/j.solener.2010.01.010. – DOI 10.1016/j.solener.2010.01.010. – ISSN 0038092X

[17] Pe~ *na–Cruz, Manuel I.*; Arancibia-Bulnes, *Camilo A.*; Iriarte-Cornejo, *Cuitlahuac*; Caba
HeliostatCharacterizationbyOpticalTechniquesandImageProcessing.In :*ASME 2012 6th Intern*
AmericanSocietyofMechanicalEngineers, *Juli*2012. – – *ISBN*978 – –0 – –7918 – –4481 – –6, 353 – –362

[18] Smith, E.J. ; Ho, C.K.: Field Demonstration of an Automated Heliostat Tracking Correction Method. In: *Energy Procedia* 49 (2014), 2201–2210. http://dx.doi.org/10.1016/j.egypro.2014.03.233. – DOI 10.1016/j.egypro.2014.03.233. – ISSN 18766102

[19] Baheti, R. ; Scott, P.: Design of self-calibrating controllers for heliostats in a solar power plant. In: *IEEE Transactions on Automatic Control* 25 (1980), Dezember, Nr. 6, 1091–1097. http://dx.doi.org/10.1109/TAC.1980.1102527. – DOI 10.1109/TAC.1980.1102527. – ISSN 0018–9286

[20] Stone, Kenneth W.: *Automatic heliostat track alignment method*. https://patents.google.com/patent/US4564275A/en. Version: Januar 1986

[21] Chen, Y. T. ; Kribus, A. ; Lim, B. H. ; Lim, C. S. ; Chong, K. K. ; Karni, J. ; Buck, R. ; Pfahl, A. ; Bligh, T. P.: Comparison of Two Sun Tracking Methods in the Application of a Heliostat Field. In: *Journal of Solar Energy Engineering* 126 (2004), Februar, Nr. 1, 638–644. http://dx.doi.org/10.1115/1.1634583. – DOI 10.1115/1.1634583. – ISSN 0199–6231, 1528–8986

[22] Chen, Y. T. ; Lim, B. H. ; Lim, C. S.: General Sun Tracking Formula for Heliostats With Arbitrarily Oriented Axes. In: *Journal of Solar Energy Engineering* 128 (2006), Mai, Nr. 2, 245–250. http://dx.doi.org/10.1115/1.2189868. – DOI 10.1115/1.2189868. – ISSN 0199–6231, 1528–8986

[23] Chong, Kok-Keong ; Wong, Chee-Woon ; Siaw, Fei-Lu ; Yew, Tiong-Keat ; Ng, See-Seng ; Liang, Meng-Suan ; Lim, Yun-Seng ; Lau, Sing-Liong: Integration of an On-Axis General Sun-Tracking Formula in the Algorithm of an Open-Loop Sun-Tracking System. In: *Sensors* 9 (2009), September, Nr. 10, 7849–7865. http://dx.doi.org/10.3390/s91007849. – DOI 10.3390/s91007849. – ISSN 1424–8220

[24] Guo, Minghuan ; Wang, Zhifeng ; Liang, Wenfeng ; Zhang, Xiliang ; Zang, Chuncheng ; Lu, Zhenwu ; Wei, Xiudong: Tracking formulas and strategies for a receiver oriented dual-axis tracking toroidal heliostat. In: *Solar Energy* 84 (2010), Juni, Nr. 6, 939–947. http://dx.doi.org/10.1016/j.solener.2010.02.015. – DOI 10.1016/j.solener.2010.02.015. – ISSN 0038092X

[25] Guo, Minghuan ; Sun, Feihu ; Wang, Zhifeng ; Zhang, Jianhan: Properties of a general azimuth–elevation tracking angle formula for a heliostat with a mirror-pivot offset and other angular errors. In: *Solar Energy* 96 (2013), Oktober, 159–167. http://dx.doi.org/10.1016/j.solener.2013.06.031. – DOI 10.1016/j.solener.2013.06.031. – ISSN 0038092X

[26] Freeman, Joshua ; Shankar, Balakrishnan ; Sundaram, Ganesh: Inverse kinematics of a dual linear actuator pitch/roll heliostat. In: *AIP Conference Proceedings* 1850 (2017), Juni, Nr. 1, 030018. http://dx.doi.org/10.1063/1.4984361. – DOI 10.1063/1.4984361. – ISSN 0094–243X. – Publisher: American Institute of Physics

[27] Grigoriev, Victor ; Milidonis, Kypros ; Blanco, Manuel: Sun tracking by heliostats with arbitrary orientation of primary and secondary axes. In: *Solar Energy* 207 (2020), September, 1384–1389. http://dx.doi.org/10.1016/j.solener.2020.07.086. – DOI 10.1016/j.solener.2020.07.086. – ISSN 0038092X

[28] Grigoriev, Victor ; Milidonis, Kypros ; Blanco, Manuel ; Constantinou, Marios: Method to determine the tracking angles of heliostats. In: *MethodsX* 8 (2021), Januar, 101244. http://dx.doi.org/10.1016/j.mex.2021.101244. – DOI 10.1016/j.mex.2021.101244. – ISSN 2215–0161

[29] Lee, Sang-Eun ; Park, Young-Chil: Modeling of Heliostat Sun Tracking Error Using Multilayered Neural Network Trained by the Extended Kalman Filter. In: *Journal of Institute of Control, Robotics and Systems* 16 (2010), Nr. 7, 711–719. http://dx.doi.org/10.5302/J.ICROS.2010.16.7.711. – DOI 10.5302/J.ICROS.2010.16.7.711. – ISSN 1976–5622. – Publisher: Institute of Control, Robotics and Systems

[30] Guangyu, Liu ; Zhongkun, Cai: Heliostat attitude angle detection method based on BP neural network. In: *MATEC Web of Conferences* 139 (2017), 00043. http://dx.doi.org/10.1051/matecconf/201713900043. – DOI 10.1051/matecconf/201713900043. – ISSN 2261–236X. – Publisher: EDP Sciences

[31] AL-Rousan, Nadia ; Mat Isa, Nor A. ; Mat Desa, Mohd K.: Efficient single and dual axis solar tracking system controllers based on adaptive neural fuzzy inference system. In: *Journal of King Saud University - Engineering Sciences* 32 (2020), November, Nr. 7, 459–469. http://dx.doi.org/10.1016/j.jksues.2020.04.004. – DOI 10.1016/j.jksues.2020.04.004. – ISSN 1018–3639

[32] Jensen, C.A. ; Reed, R.D. ; Marks, R.J. ; El-Sharkawi, M.A. ; Jung, Jae-Byung ; Miyamoto, R.T. ; Anderson, G.M. ; Eggen, C.J.: Inversion of feedforward neural networks: algorithms and applications. In: *Proceedings of the IEEE* 87 (1999), September, Nr. 9, S. 1536–1549. http://dx.doi.org/10.1109/5.784232. – DOI 10.1109/5.784232. – ISSN 1558–2256. – Conference Name: Proceedings of the IEEE

[33] Paszke, Adam ; Gross, Sam ; Chintala, Soumith ; Chanan, Gregory ; Yang, Edward ; DeVito, Zachary ; Lin, Zeming ; Desmaison, Alban ; Antiga, Luca ; Lerer, Adam: Automatic differentiation in PyTorch. (2017), Oktober. https://openreview.net/forum?id=BJJsrmfCZ

[34] Yang, Yuzhe ; Zha, Kaiwen ; Chen, Yingcong ; Wang, Hao ; Katabi, Dina: Delving into Deep Imbalanced Regression. In: *Proceedings of the 38th International Conference on Machine Learning*, PMLR, Juli 2021, 11842–11851. – ISSN: 2640-3498

[35] Branco, Paula ; Torgo, Luís ; Ribeiro, Rita P.: A Survey of Predictive Modeling on Imbalanced Domains. In: *ACM Computing Surveys* 49 (2016), August, Nr. 2, 31:1–31:50. http://dx.doi.org/10.1145/2907070. – DOI 10.1145/2907070. – ISSN 0360–0300

[36] Fieguth, Paul: Distance-Based Classification. In: *An Introduction to Pattern Recognition and Machine Learning*. Cham, Switzerland : Springer Nature Switzerland, 2022. – ISBN 978–3–030–95995–1, S. 117–132

[37] Versloot, Christian: *Machine learning articles*. https://github.com/christianversloot/machine-learning-articles/blob/6b0d8425bbb5b1dd28618d2c5dfdaf2a24f63513/how-to-use-l1-l2-and-elastic-net-regular md. Version: Februar 2023. – original-date: 2022-02-15T18:12:31Z

[38] Klambauer, Günter ; Unterthiner, Thomas ; Mayr, Andreas ; Hochreiter, Sepp: Self-Normalizing Neural Networks. In: *Advances in Neural Information Processing Systems* Bd. 30, Curran Associates, Inc., 2017

[39] Khalsa, Siri Sahib S. ; Ho, Clifford K. ; Andraka, Charles E.: AN AUTOMATED METHOD TO CORRECT HELIOSTAT TRACKING ERRORS.

[40] Malan, K. ; Gauché, P.: Model based Open-loop Correction of Heliostat Tracking Errors. In: *Energy Procedia* 49 (2014), Januar, 2118–2124. http://dx.doi.org/10.1016/j.egypro.2014.03.224. – DOI 10.1016/j.egypro.2014.03.224. – ISSN 1876–6102

[41] Pargmann, Max ; Leibauer, Moritz ; Nettelroth, Vincent ; Maldonado Quinto, Daniel ; Pitz-Paal, Robert: *It is Not About Time - A New Standard for Open-Loop Heliostat Calibration Methods*. http://dx.doi.org/10.21203/rs.3.rs-2898838/v1. Version: Mai 2023

[42] Pargmann, Max ; Leibauer, Moritz ; Nettelroth, Vincent ; Maldonado Quinto, Daniel ; Pitz-Paal, Robert: *Enhancing Heliostat Calibration on Low Data by Fusing Robotic Rigid Body Kinematics with Neural Networks*. http://dx.doi.org/10.21203/rs.3.rs-2921011/v1. Version: Mai 2023

A. Heliostat Parameters Table

Heliostat Kinematic Model Geometry Parameters (Part 1)			
Parameter Name	**Type**	**Scaling**	**Purpose**
First Joint Translation AZ	FLoat	1rad	Azimuth component of the polar coordinates translation from base to first joint.
First Joint Translation EL	FLoat	1rad	Elevation component of the polar coordinates translation from base to first joint.
First Joint Translation Rad	FLoat	1m	Radius component of the polar coordinates translation from base to first joint.
First Joint Rotation East	FLoat	1rad	First joint tilt around the x (east) axis.
First Joint Rotation North	FLoat	1rad	First joint tilt around the y (north) axis.
First Joint Rotation Up	FLoat	1rad	First joint tilt around the z (up) axis.
Second Joint Translation AZ	FLoat	1rad	Azimuth component of the polar coordinates translation from first to second joint.
Second Joint Translation EL	FLoat	1rad	Elevation component of the polar coordinates translation from first to second joint.
Second Joint Translation Rad	FLoat	1m	Radius component of the polar coordinates translation from first to second joint.
Second Joint Rotation East	FLoat	1rad	Second joint tilt around the x (east) axis.
Second Joint Rotation North	FLoat	1rad	Second joint tilt around the y (north) axis.
Second Joint Rotation Up	FLoat	1rad	Second joint tilt around the z (up) axis.
Concentrator Translation AZ	FLoat	1rad	Azimuth component of the polar coordinates translation from second joint to concentrator center.
Concentrator Translation EL	FLoat	1rad	Elevation component of the polar coordinates translation from second joint to concentrator center.
Concentrator Translation Rad	FLoat	1m	Radius component of the polar coordinates translation from second joint to concentrator center.

AZ = Azimuth, EL = Elevation, Rad = Radius

Heliostat Kinematic Model Geometry Parameters (Part 2)			
Parameter Name	**Type**	**Scaling**	**Purpose**
Concentrator Rotation East	FLoat	1rad	Concentrator tilt around the x (east) axis.
Concentrator Rotation North	FLoat	1rad	Concentrator tilt around the y (north) axis.
Concentrator Rotation Up	FLoat	1rad	Concentrator tilt around the z (up) axis.
Actuator 1 Increment	FLoat	1%	Actuator 1 stroke length change per actuator step.
Actuator 1 Stroke Length	FLoat	1%	Actuator 1 initial stroke length.
Actuator 1 Offset	FLoat	1%	Offset between actuator 1 and the first joint's pivoting point.
Actuator 1 Radius	FLoat	1%	Offset between actuator 1 working point and the first joint's pivoting point
Actuator 1 Initial Angle	FLoat	1rad	Actuator 1 initially induced angle.
Actuator 2 Increment	FLoat	1%	Actuator 2 stroke length change per actuator step.
Actuator 2 Stroke Length	FLoat	1%	Actuator 2 initial stroke length.
Actuator 2 Offset	FLoat	1%	Offset between actuator 2 and the first joint's pivoting point.
Actuator 2 Radius	FLoat	1%	Offset between actuator 2 working point and the first joint's pivoting point
Actuator 2 Initial Angle	FLoat	1rad	Actuator 2 initially induced angle.

AZ = Azimuth, EL = Elevation, Rad = Radius