Institute of Solar Research

Institute of Metal Forming

RWTH AACHEN UNIVERSITY

Chair of Material Modelling in Forming Technology

Master Thesis

Vibration Analysis on Flexible Pipe Connectors of Parabolic Trough Solar Collectors for Predictive Maintenance

By

Irene Biermann

418294

Conducted at the Institute of Solar Research of the German Aerospace Center

From 08.01.2025 to 24.06.2025

First Examiner: Univ.-Prof. Dr.-Ing. Sebastian Münstermann

Second Examiner: Dipl.-Inform. Franzke Martin

Supervisors: Dr. Sonja Kallio, Dr. Benedikt Kölsch, Niklas Fehlemann M.Sc.

Almeria, 24.06.2025

Abstract

Parabolic Trough Collectors (PTC) are the most proven concentrating solar power (CSP) technology. The part of a PTC that is typically the first to fail is the rotation and expansion performing assembly (REPA), which is a flexible pipe connection between the movable receiver tube and the stationary field tubing. The combination of high temperatures, pressures and mechanical loads due to the movement makes this component prone to failure. A malfunction of the REPA can lead to a leakage of the hot heat transfer fluid, potentially causing fires and shut downs of the collector row or the whole power plant.

The aim of this work is to develop a predictive maintenance strategy based on the vibrations that occur in the REPAs. A previous work investigated the change in the vibration patterns over the life cycle of REPAs in PTCs for condition monitoring [1]. This thesis expands on this idea by using the change in vibrations to predict the REPAs future malfunctions.

To distinguish between the vibrations of a normally functioning REPA and a REPA shortly before a leak occurs, a convolutional neural network is trained for classification. The vibration data used for training is collected in an accelerated life cycle test campaign and analyzed using short time Fourier transform. Two kinds of REPA are tested, first a rotary flex hose assembly and later a ball joint assembly.

By employing a machine learning classification model, REPA malfunctions were predicted at least 1000 cycles before they occurred, which translates to more than 2.5 years of operation. This proves, that installing vibration sensors in PTC power plants can help making the operation safer and more cost effective.

Contents

Acronyms

PSA Platforma Solar de Almeria

CIEMAT Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas

CSP Concentrating Solar Power

HTF Heat-Transfer Fluid

DNI Direct Normal Irradiation

PTC Parabolic Trough Collector

REPA Rotation and Expansion Performing Assembly

RFHA Rotary Flex Hose Assembly

BJA Ball Joint Assembly

R2F Run to Failure

PvM Preventive Maintenance

PdM Predictive Maintenance

STFT Short Time Fourier Transform

TP True Positive

FN False Negative

FP False Positive

TN True Negative

ANN Artificial Neural Network

MSE Mean Squared Error

(W)BCE Loss Weighted Binary Cross Entropy Loss

KL Divergence Kullback-Leibler Divergence

ReLU Rectified Linear Unit Function

SGD Stochastic Gradient Descent

Adam Adaptive Moment Estimation

RMSprop Root Mean Square Propagation

CNN Convolutional Neural Network

SCADA Supervisory Control and Data Acquisition System

List of Figures

List of Tables

1 Introduction

In 2016, 196 countries signed the Paris agreement. Its goal is, to limit the global average temperature to 1.5℃ above the pre-industrials levels [2]. In order to achieve this, the global greenhouse gas emissions need to be cut by 50% by 2030 and reach net zero by 2050. With 13,800 million tonnes of CO2 being released in 2024, the energy sector has the highest contribution to global emissions out of all of the emissions sectors [3]. Simultaneously, the global power demand is rising, with electricity consumption rising by 4.3% in 2024 and an expected growth of 3.9% per year until 2027 [3].

These factors stress the need for low-emission energy sources to uphold the Paris agreement and to avoid severe impacts of the climate change, such as droughts, heatwaves and floods due to extreme rainfalls. Solar photovoltaic was the fastest growing low-emission energy source in 2024 and is predicted to become the largest renewable energy source by 2029 [4]. However, the availability of solar energy is intermittent due to day/night cycles and varying weather conditions.

Concentrating solar power (CSP) offers an alternative technology to transfer solar energy to electrical energy with the possibility of energy storage in the form of molten salts, allowing for electricity generation at night. The most common CSP technology is the parabolic trough collector (PTC), where parabolic mirrors are used to concentrate the direct solar irradiation onto a receiver tube that contains a heat transfer fluid (HTF). The hot HTF is used to drive a heat engine to generate electricity.

The part of a PTC that is typically the first to fail is the rotation and expansion performing assembly (REPA), which is a flexible pipe connection between the movable receiver tube and the stationary field tubing. The combination of high temperatures, pressures and mechanical loads due to the movement makes this component prone to failure. A malfunction of the REPA can lead to a leakage of the hot heat transfer fluid, causing shut down of the plant partly or fully or potentially even fires.

The aim of this work is to develop a predictive maintenance strategy based on the vibrations that occur in the REPAs. A previous work investigated the change in the vibration patterns over the life cycle of REPAs in PTCs for condition monitoring [1]. This thesis expands on this idea by using the change in vibrations to predict the REPAs future malfunctions. Similar approaches have been employed for predictive maintenance of

several different components, e.g. wind turbines or the journal bearings in hydropower plants [5] [6].

To establish a predictive maintenance strategy, vibration data of an entire REPA life cycle is required. Since a REPA is usually replaced after 25 to 30 years, it is no practical to collect the vibration data on a real PTC. Instead, a test rig is used to perform accelerated rotation cycles. Here, it is possible to perform around 570 rotation cycles per day, making it possible to reach 10,000 cycles in under three weeks. Two REPAs are tested simultaneously. The test rig simulates the conditions the REPAs would be exposed to, such as high temperatures and pressure. It is located at the *Platforma Solar de Almeria* (PSA) in Spain. It is owned by the Spanish research institution *Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas* (CIEMAT). The PSA is the largest research center for concentrated solar power research, providing state-of-the-art testing facilities for both components and systems in solar thermal energy. Its research spans several critical areas, including the enhancement of heat storage systems, the creation of new collector technologies, and the examination of material performance under harsh operational conditions.

The vibration data obtained by the accelerated life cycle tests is analyzed using Fourier transform techniques to calculate spectrograms of the vibrations occuring during each vibration cycle. Afterwards, a machine learning model is trained to classify between spectrograms of cycles that happened close to a malfunction of one of the REPAs and ones where the REPAs functioned normally. This model could later be used to analyze vibrations while the REPA is still in service to plan maintenance. This would increase the reliability of PTC power plants and reduce their operation and maintenance costs.

2 State of the Art in Science and Technology

As the global demand for clean, reliable energy grows, concentrating solar power (CSP) grows in importance for the energy mix [7]. Unlike photovoltaic systems, CSP systems can directly integrate thermal energy storage, allowing them to supply power even when the sun is not shining. This makes CSP valuable for stabilizing energy grids and supporting the transition away from fossil fuels. The working principle of CSP techniques and parabolic trough collectors in particular will be discussed in Chapter 2.1.
The goal of this work is to develop a predictive maintenance strategy for the flexible pipe connectors in parabolic trough collectors by analyzing the vibrations occurring in them. The fundamentals of predictive maintenance are the topic of Chapter 2.2, while Chapter 2.3 delves into the working principle of the sensors capturing the vibration data. In Chapter 2.4 the techniques used in vibration analysis will be discussed. Chapter 2.5 delves into machine learning for classification tasks, which is used for predictive maintenance.

2.1 Concentrating Solar Power

CSP systems use the irradiation of the sun to generate electrical energy. They use optical systems such as mirrors or lenses to concentrate solar rays onto a receiver, where the solar energy is converted into heat, which can then be used to drive a thermal engine. CSP systems generally consist of five steps: concentration, absorption, transfer, storage and generation [8].
In the first step, the solar irradiation of a large area is concentrated onto a smaller receiver. The concentrated light is absorbed by the receiver and turned into heat. The larger the ratio of the collection area to the area of the receiver surface is, the higher the achieved temperature in the receiver becomes. This is also called the *concentration ratio* of the CSP system. A heat-transfer fluid (HTF) is used to transfer the heat away from the absorber. It is possible to store the heat in a thermal energy storage to use it at a later time. While this step is optional, the potential for energy storage results in a significant advantage over other renewable energy sources such as photovoltaic[7]. In the last step, a heat engine is used to generate electricity [8].

Since the sun is not in a fixed position but moves across the sky, the CSP system has to be moved to track the sun's movement. The progression of the sun can be divided into a daily east-west movement and a north-south movement that happens both daily and seasonally. There are two subsets of CSP technologies: point-focusing and line-focusing systems. Point-focusing systems employ a two-axis tracking system that tracks both the east-west and the north-south movement. This results in a higher concentration ratio, however the solar tracking is more complex and therefore more expensive to implement. Line-focusing systems only track in the east-west direction, since most of the daily sun's progression happens on this axis. The one-axis tracking allows for a simpler implementation, thus making it cheaper. The achieved concentration ratios are lower than thise in the point-focusing systems, making line-focusing systems less efficient [8].

The collecting surface of the CSP system can either be continuous or consist of several discrete facets. The continuous systems have the advantage, that there is no loss of sunlight in between the mirror surfaces which results in higher concentration ratios. However, the size of the receiver is limited since it has to be mounted with the collector and follow the same tracking motion. Discrete systems on the other hand lose some sunlight in the gaps between the mirrors but allow for a larger receiver that can be stationary.

The distinction between point-focusing and line-focusing and the distinction between continuous and discrete systems allow for four different types of CSP systems, of which Figure 2.1 shows an overview: continuous line-focusing (parabolic trough collectors), discrete line-focusing (linear Fresnel reflectors), continuous point-focusing (parabolic dish reflectors) and discrete point-focusing (heliostat fields). [8]

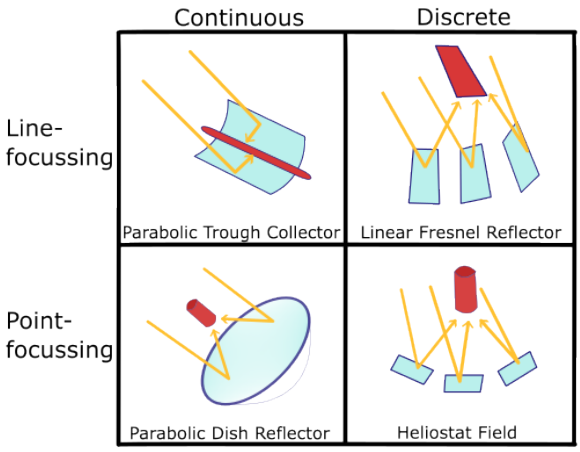

Figure 2.1: Overview of the different CSP technologies, red: receiver, blue: mirror panels

In contrast to photovoltaic panels, CSP systems can only efficiently use the direct irradiation of the sun. This means, that CSP is only suitable for regions with low cloud coverage. The direct normal irradiation (DNI) quantifies the solar normal irradiance on the surface of the earth excluding radiation that has been scattered or reflected. The amount of scattered radiation is dependent on the elevation angle of the sun, cloud coverage, humidity of the air and other particles in the air. Figure 2.2 shows a map of the DNI in different parts of the world. The red areas in the map are more suitable for electricity generation with CSP [9].

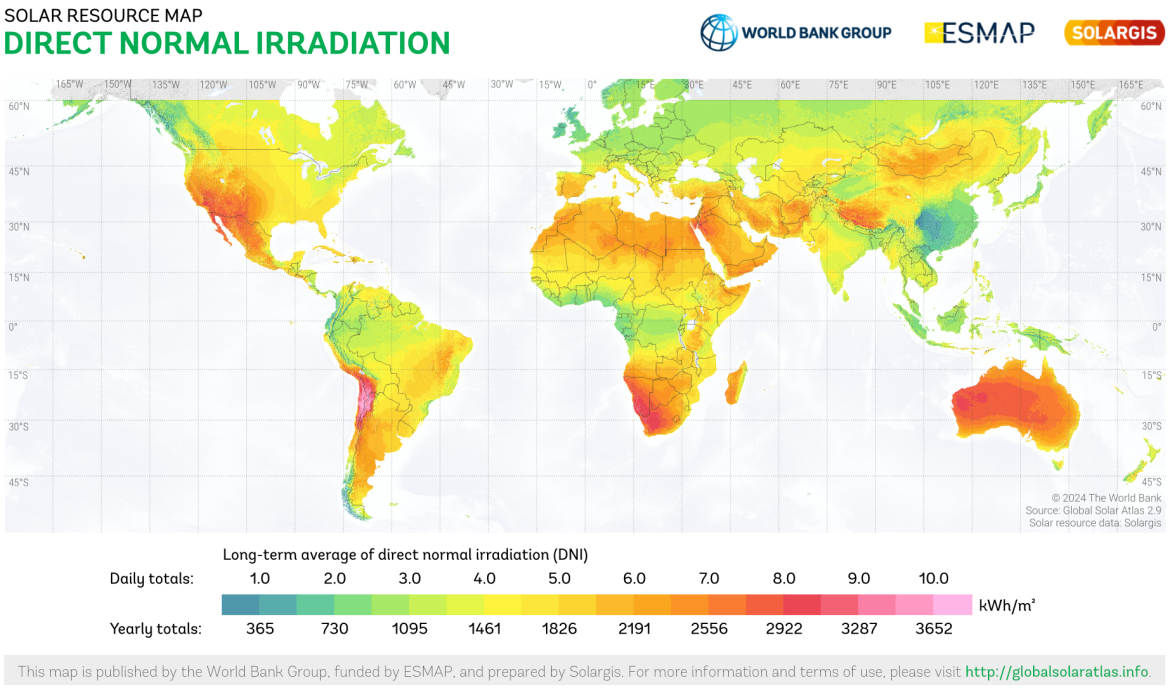

Figure 2.2: DNI map of the world [9]

2.1.1 Parabolic Trough Collectors

The CSP technology considered in this work is the Parabolic Trough Collector (PTC). The assembly of a PTC is shown schematically in Figure 2.3.

The concentrators used in PTCs are parabolic shaped reflectors which are made of aluminum or silvered-glass mirrors. The receiver is a tube containing the heat transfer liquid which is placed in the focal axis of the concentrator, absorbing the reflected radiation. In order to minimize heat loss, the receiver tube is encapsulated in an evacuated glass cover which has a non-reflective coating that results in a very high transmissivity of the cover. The connection of the tube and the glass cover is realized by metallic bel-

lows, which achieve the vacuum-tight enclosure and also compensate the differences in thermal expansion between both components. [10]

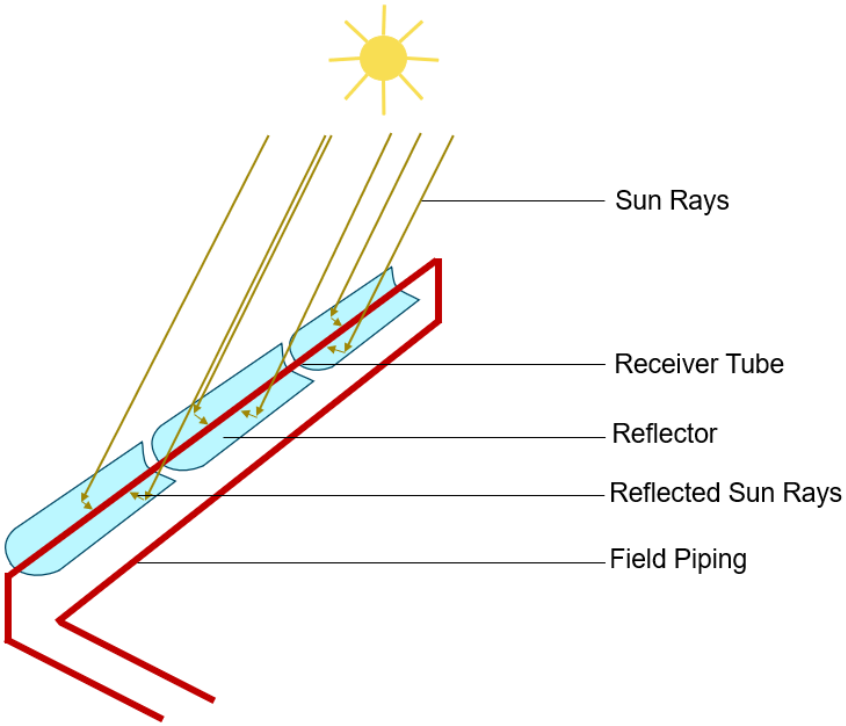

Figure 2.3: Assembly of a parabolic trough collector

To maximize the energy collected by the system, the concentrators are rotated along a single axis during the day to follow the track of the sun. [11] The collectors can be aligned along the north-south axis or along the east-west axis. If the north-south axis is chosen, the sun has to be tracked along the east-west axis, whereas a PTC that is aligned with the east-west axis tracks the sun along the north-south direction.
Figure 2.4 shows the rotation movement that is daily performed by the collector. When the sun does not shine at night, the PTCs rests at α_{stow}. Before the sun rises, the collector is moved to α_{start} in a quick and continuous motion. After the sun rises, the collector tracks is movement from α_{start} to the sunset position α_{end}. This movement does not happen continuous but in small increments every 20 to 40 seconds. To determine the current position of the sun, approximating algorithms are used. At the end of the day, the collector is moved back to α_{stow}. This movement is again fast and continuous. [11]

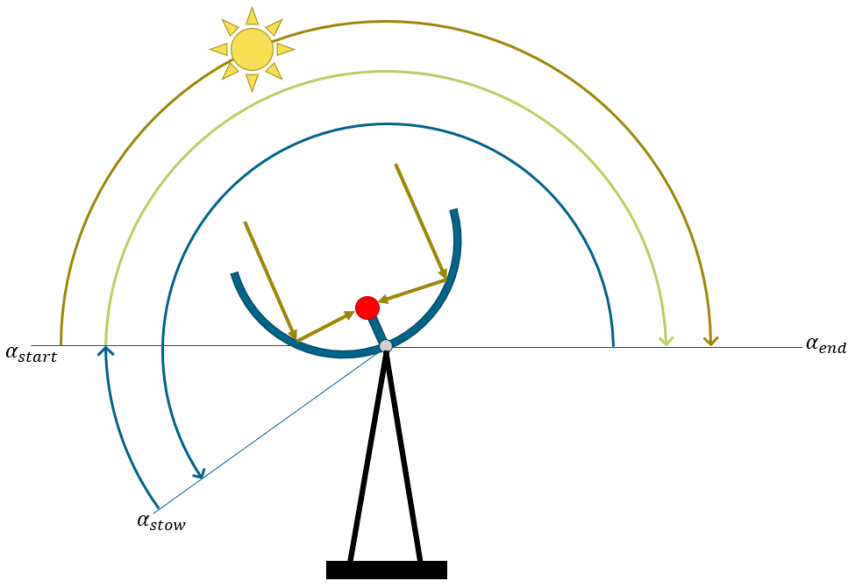

Figure 2.4: Daily sun-tracking movement of the PTC(green: slow, incremental motion, blue: fast, continuous motion)

An important aspect in the design of a PTC system is the choice of heat transfer fluid. It determines the operational temperature that can be achieved in the PTC and the maximum power cycle efficiency that can be achieved [12]. There are several different HTF available for usage in PTC power plants.

Synthetic oil are the most widely used HTFs in parabolic trough systems. Synthetic oils, such as Therminol VP-1 and Helisol, have high thermal stability and are able to operate at high temperatures up to around 400 °C. They offer good heat transfer properties and are relatively easy to handle, making them a popular choice in commercial solar power plants. [12]

Molten Salts, a mixture of sodium nitrate and potassium nitrate, is often used in high-temperature systems. Molten salt is advantageous because it can store thermal energy efficiently, making it ideal for systems requiring thermal storage. It operates at temperatures up to 565 °C and can retain heat for extended periods, allowing solar plants to produce electricity even when sunlight is not available. However, they can be highly corrosive, damaging the receiver tubes. Additionally, if there are cold spots in the receiver (e.g. due to clouds), there is the danger of the salt solidifying.[12]

In some applications, water or steam can be used as an HTF, particularly in lower-temperature systems. Water is abundant and inexpensive, but it requires higher pressure to operate at elevated temperatures and is limited in the temperature range it can effectively handle (up to about 300 °C). [12]

Although less common, air can also serve as an HTF, especially in systems designed

for lower temperatures. It is non-toxic, inexpensive, and readily available, but its heat transfer properties are less effective than those of liquids like oils or molten salts. [12]

2.1.2 Rotation and Expansion Performing Assemblies

Since the collector surface of a PTC is rotated to track the movement of the sun, the receiver tube has to be moved in order to stay in the focal line of the receiver. This movable part of the tube has to be connected to the field piping with a flexible pipe connector. Additionally, the thermal expansion of the receiver tube results in a translational movement that has to be absorbed by the pipe connector. The translational movement follows a daily pattern, where the tube expands in the morning when the temperature of the HTF rises and contracts again in the evening when the temperature falls. The pipe connectors that compensate for these two movements are known as rotation and expansion performing assemblies (REPAs).
Per PTC, two REPAs are required, one on each side of the collector. The REPAs have to withstand cyclic loads in addition to temperatures of about 400 °C and pressures of up to 35 bar, making the REPAs prone to failure. They have an estimated lifetime of 10,000 cycles, which corresponds to 25-30 years with one cycle per day. [13]
Failure of the REPAs can result in leakage of the HTF, which can lead to fires due to the high temperature of the fluid. Furthermore, after a failure maintenance is required resulting in unplanned downtime for the PTC power plant. [13]
There are two main types of REPAs that are used in PTC power plants: rotary flex hose assemblies and ball joint assemblies. They will be presented in the following sections. [14]

Rotary Flex Hose Assemblies

Rotary flex hose assemblies (RFHAs) consist of a flexible metal hose, a swivel joint and a torque transmitter, also known as a torque sword. Figure 2.5 shows the schematic layout of a RFHA.
The flexibility of the metal hose is provided by its corrugated structure along its length. Additionally, it has a thermal insulation to reduce heat loss. The metal hose compen-

sates the translational movement caused by the thermal expansion of the receiver tube, while the swivel joint compensates for the rotational movement. The swivel joint has one rotational degree of freedom and connects the fixed pipe to the metal hose. The torque sword transmits the rotational movement and connects the PTC to the swivel joint.[15]

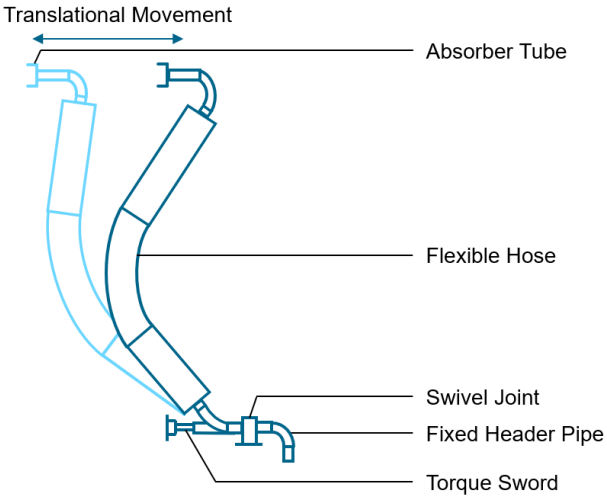

Figure 2.5: Schematic layout of a REPA

Ball Joint Assemblies

A ball joint assembly (BJA) is made up of three ball joints. Figure 2.6 shows the schematic layout of one ball joint. A ball joint consists of a ball, which is a pipe with a spherical end, and a housing for this ball. Two compression seals keep this setup in place. Through a fill coupling in the housing, a sealant can be injected. Graphite is often used as a sealant. Since the sealant can be refilled the joint is serviceable, making the BJA more popular in commercial plants than RFHAs. [14]
Each ball joint has three rotational degrees of freedom, making it possible for the arrangement to compensate for both the rotational and the translational movement. The first ball joint is attached to the receiver tube and another is attached to the fixed field tubing. The third ball joint is placed between the other two. The joints are connected by pipe sections. [14]

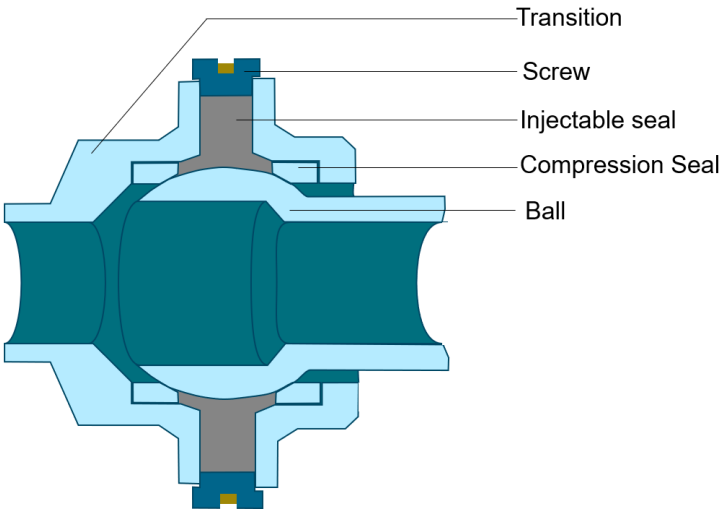

Figure 2.6: Schematic layout of a ball joint

2.2 Maintenance Strategies

Maintenance is a critical aspect of any operation, particularly in industries relying on machinery and equipment to function efficiently. There are several maintenance strategies that can be adopted to ensure that systems operate normally. The goal is to minimize downtime and repair costs while also ensuring a safe operation. Common strategies are run to failure (R2F), preventive maintenance (PvM), and predictive maintenance (PdM) [16]. Each strategy has its advantages and disadvantages, but in recent years, predictive maintenance has gained significant attention, since anticipating failure before it occurs can minimize the systems downtime needed for repair and makes prior planning of the repairs possible [17].

The Run to Failure strategy is the simplest but also the least effective one. In this strategy, equipment is allowed to operate until it breaks down. There are no scheduled maintenance activities or efforts to prevent failures before they occur. Once equipment fails, it is repaired or replaced. This method is typically employed in low-risk situations where the cost of downtime is negligible or the consequences of failure are minimal. However, the major downside of R2F is the potential for costly unplanned downtime, expensive repairs, and the unexpected consequences of a failure. It is often considered inefficient for high-value assets, as it does not maximize equipment lifespan or operational efficiency [16].

Preventive Maintenance aims to prevent equipment failures by scheduling regular inspections, servicing, and repairs. This strategy is either based on a fixed time interval, such as performing maintenance tasks every month, quarter, or year, regardless of the

equipment's actual condition, or on a set number of process iterations. The key benefit of PvM is that it helps avoid the worst-case scenario of unexpected breakdowns, and it can extend the lifespan of equipment through consistent care. However, it also results in downtime of the system that might not be necessary which can be costly. Additionally, PvM does not provide insight into the actual health of the equipment, meaning some components may undergo unnecessary maintenance, while others could fail prematurely due to insufficient attention.[16][17]

Predictive Maintenance represents a more advanced and data-driven approach to maintaining equipment. It focuses on monitoring the actual condition of equipment in real-time, often using sensors and non-destructive testing technologies, such as vibration analysis, infrared thermography, and oil analysis. PdM predicts potential failures before they occur, allowing maintenance activities to be planned just in time, reducing unplanned downtime and avoiding unnecessary repairs. This method stands in contrast to PvM, where maintenance may occur regardless of the equipment's actual condition.[16]

The predictive aspect of PdM stems from its ability to forecast the future condition of equipment, using statistical process control principles to determine when maintenance should take place. This ensures that interventions are both cost-effective and timely, avoiding both premature maintenance and unexpected breakdowns. Additionally, PdM minimizes system disruptions by enabling maintenance to be performed while equipment remains in service, thus reducing the operational impact.[17]

One of the key benefits of PdM is its ability to integrate with computerized maintenance management systems. This integration allows for efficient scheduling and execution of maintenance tasks based on the condition data collected, ensuring that the right actions are taken at the right time. For large-scale facilities, this system integration is critical, as it helps manage the maintenance of thousands of pieces of equipment without overwhelming the workforce.[16]

The adoption of PdM leads to substantial cost savings and improved system reliability. For example, industries like energy production benefit not only from reduced downtime and lower repair costs but also from avoiding fines related to non-delivery. By forecasting equipment failures before they disrupt operations, PdM improves profitability and operational continuity, contributing to long-term financial and operational benefits.[17]

One of the main advantages of PdM is that it optimizes the maintenance process by reducing the frequency of unnecessary maintenance tasks. Since maintenance is

only performed when required, resources are used more effectively, and equipment operates more efficiently. Additionally, the ability to predict failures before they happen leads to reduced downtime, increased productivity, and lower overall maintenance costs. However, implementing a predictive maintenance program can be resource-intensive initially. It requires an upfront investment in sensors, data analytics platforms, and possibly a change in organizational practices to collect and interpret data properly. Nevertheless, for industries where equipment reliability is critical, such as manufacturing, energy production, and transportation, the long-term savings and improved operational efficiency often outweigh these initial costs.[17]

2.3 Vibration Sensors

The aim of this work is to develop a predictive maintenance strategy based on the vibrations occurring in the REPAs of PTCs. In order to do this, vibration measurements of the REPA's lifecycles are needed. The vibrations are recorded using accelerometers.

Accelerometers measure an object's proper acceleration. This is the acceleration relative to an observer that is in free fall [18]. In contrast to coordinate acceleration, which is relative to a coordinate system, this is not dependent on the observer's choice. Thus, an accelerometer that does not move relative to the earth's surface will measure an acceleration of 1g upwards, which must be subtracted from the accelerometer's measurements. This is due to Einstein's equivalence principle, stating that gravitation causes the same effects on an object as an acceleration and the two can therefore not be distinguished [19]. In free fall, the accelerometer will not measure any acceleration, as the observer is also in free fall.

There are several types of mechanical accelerometers, for example electrical, capacitive or piezoelectric accelerometers [20]. In this work, piezoelectric accelerometers are used, which will be explained in the following paragraphs.

Piezoelectric accelerometers use the piezoelectric effect to measure acceleration by employing piezoelectric materials. Their advantages include a high sensitivity at high frequencies and a good resistance to high temperatures. Additionally, they can be produced to be light and small making them suitable for many applications.

The piezoelectric effect refers to the phenomenon, where mechanical stress causes

accumulation of the electric charge in a crystal, resulting in an electric field [21]. This effect is reversible, meaning that an electric field leads to a change in the dimension of the piezoelectric crystal, known as the converse piezoelectric effect [22].

The effect is caused by electric dipole moments that occur in solids. The dipole moments can either be induced or can be carried by dipolar molecule groups. Each dipole can be denoted as a vector. The dipole density $\vec{P}$ describes all dipoles in a material. In a piezoelectric materials, external stress results in a re-orientation of the dipoles, either by changing the dipole-inducing surrounding or the orientation of the molecules carrying the dipoles. The change in the orientation of dipoles leads to a change in $\vec{P}$, either in its strength or its orientation or both. From the outside, a change in $\vec{P}$ can be observed as a change in the surface charge density on the crystal faces, resulting in a changed electric field. [22]

Piezoelectric behavior can be observed in crystalline materials without an inversion axis, for example quartz. Ceramic materials with randomly oriented grains also need to exhibit ferroelectricity in order to have piezoelectric properties. The most common ceramic piezoelectric material is lead zirconate titanate, also known as PZT. Single crystals have a long life span in which the sensitivity remains high, however their sensitivity is generally lower than that of ceramic piezoelectric materials. While ceramics offer a high sensitivity, it degrades over time, resulting in a shorter lifespan of piezoelectric ceramics compared to single-crystal materials. [22]

Piezoelectric materials have many applications, including portable lighters, scanning probe microscopy and various sensors. The sensors have in common, that they use the piezoelectric effect to turn a mechanical load into an electric signal. They can be used to detect e.g. temperature, strain or pressure. In this work they are utilized as accelerometers. [23]

A piezoelectric accelerometer typically consists of a sensing crystal, which is attached to a seismic mass. To ensure linear behavior and to make a rigid structure, a preload ring is placed around the sensing crystal. When the sensor is accelerated, the seismic mass puts stress on the sensing crystal, causing an electrical output by the piezoelectric effect. The electrical output is collected on electrodes and further transmitted by wires. Figure 2.7 shows the typical layout of an accelerometer. [24] [23]

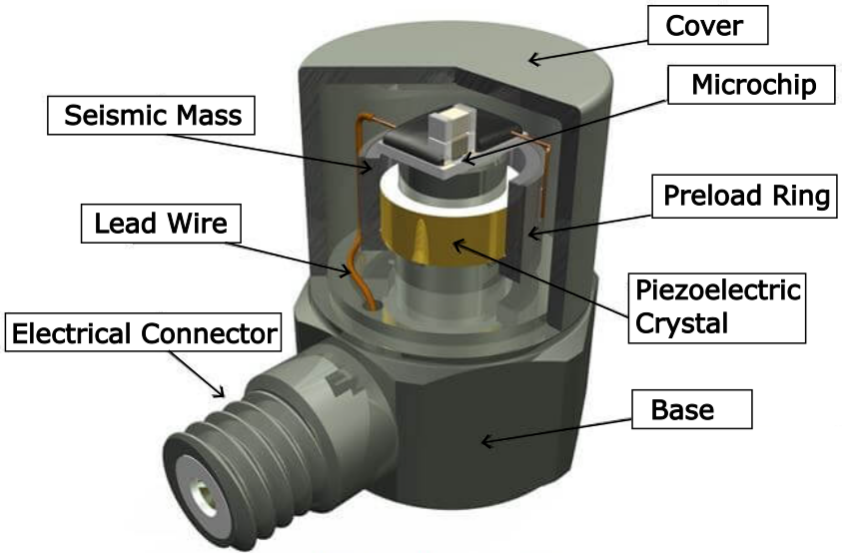

Figure 2.7: Typical layout of an accelerometer [24]

2.4 Vibration Analysis

In order to analyze the vibrations occurring in the flexible pipe connectors it is helpful to examine which frequencies are present in the signal and which energies they have. This Chapter introduces frequency-domain methods used in vibration analysis, beginning with the Fourier transform in section 2.4.1, which provides a frequency spectrum for time-dependent signals. In the following section 2.4.2 the discrete Fourier transform, a practical approach for analyzing sampled data, is introduced. Finally, this Chapter covers the short-time Fourier transform in section 2.4.3, which extends the discrete Fourier transform by providing time-resolved frequency information.

2.4.1 Fourier Transform

The Fourier Transform is a technique that allows for the representation of a time-dependent function in the frequency domain. It was first proposed by Fourier in 1822, who found that it is possible to expand any function into a series of sines [25].
The general definition of the Fourier transform is given by Equation 2.1, where *f* denotes the frequency and *t* represents the time. The original time-dependent signal is given by *s(t)* and is transformed into the frequency spectrum S(f). The transformation can be inversed by applying Equation 2.2 which turns the frequency spectrum into the

time-dependent signal. [26]

$$S(f) = \int_{-\infty}^{\infty} s(t) * \mathrm{e}^{-i2\pi ft}\mathrm{d}t, \qquad \forall f \in \mathbb{R} \tag{2.1}$$

$$s(t) = \int_{-\infty}^{\infty} S(f) * \mathrm{e}^{i2\pi ft}\mathrm{d}f, \qquad \forall t \in \mathbb{R} \tag{2.2}$$

A simple example of a Fourier transform is given in Figure 2.8. The left side of the Figure shows a time-dependent signal $s(t) = 2 * \sin(2\pi * 4t) + 5 * \sin(2\pi * 2t) + 3 * \sin(2\pi * 0.5t)$. The signal only consists of three different components, however the plot already looks complex. On the right side of the Figure the Fourier transform of the signal is shown. It shows the three frequencies that can be found in the signal s(t) and the amplitude at which these frequencies appear. The result of the transform is called the spectrum of the signal.

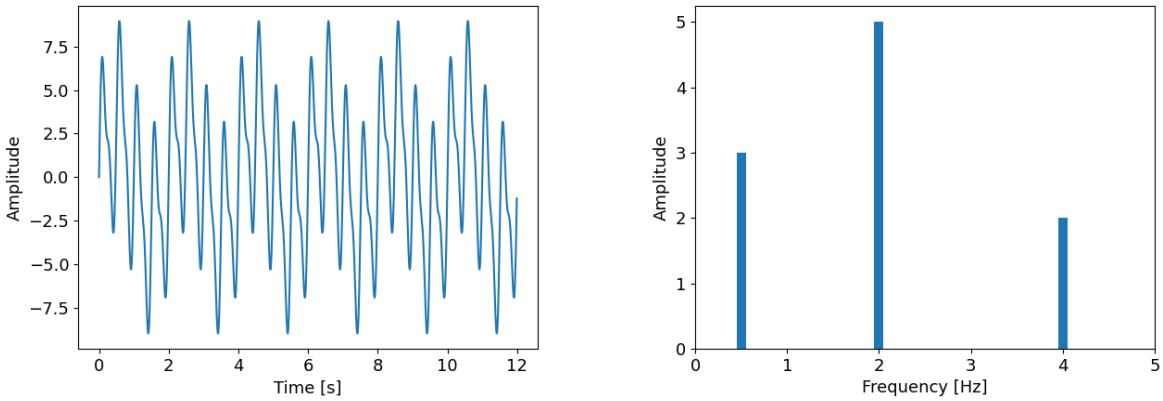

Figure 2.8: Left: A simple example signal, Right: Fourier transform of the signal

The length of the analyzed signal influences the quality of the Fourier transform. If the signal length is not an integer multiple of the period length of the sinus waves, the Fourier transform recognizes frequency components that are not actually present in the signal. If there are frequencies in the spectrum that do not appear in the original signal this is call *spectal leakage* [27]. To illustrate the effect this has on the quality of the spectrum, Figure 2.9 shows the result of the Fourier transform of the above signal with a signal length of 12.3s.

More broadly, spectral leakage can be viewed as an effect of windowing, which is the convolution of the signal with a window function. The finite signal length can then be interpreted as the result of the multiplication of the signal with a rectangular window.[27]

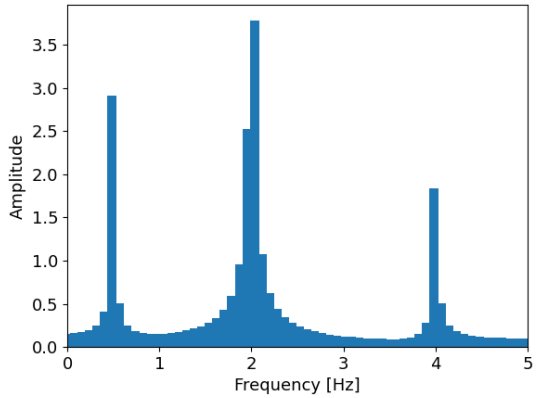

Figure 2.9: Fourier transform of 12.3 seconds of the signal in Figure 2.8 (left) showing spectral leakage

2.4.2 Discrete Fourier Transform

Since in the application of vibration analysis, the signal is not continuous but consists of discrete data points, the approach explained in section 2.4.1 has to be modified. The solution for this problem is called the *discrete Fourier transform*. It is defined by Equation 2.3, where $\{x_n\} = x_0, x_1, ..., x_n$ and $\{X_n\} = X_0, X_1, ..., X_{N-1}$ are two sequences of *N* complex numbers. The inverse transformation is given by Equation 2.4. [28]
The frequency at which the discrete data points are measured is called the sampling frequency f_s. The highest frequency that can be resolved by the Fourier transform is $f_s/2$, also known as the Nyquist frequency. If there are frequencies above the Nyquist frequency, they appear as a lower frequency in the Fourier spectrum. This effect is known as *aliasing*. [28][27]
To avoid aliasing, two main strategies are used. The first one is two increase the sampling rate so that it is more than twice the highest frequency present in the signal. This ensures that all frequency components are below the Nyquist limit. The second strategy is to apply an anti-aliasing filter before sampling, which is a low-pass filter that removes all frequency components above the Nyquist frequency, This guarantees, that no out-of-range frequencies are present when the signal is digitized. [27]

$$X_k = \sum_{n=0}^{N-1} x_n * \mathrm{e}^{-i2\pi \frac{k}{N} n} \tag{2.3}$$

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k * \mathrm{e}^{i2\pi \frac{k}{N} n} \tag{2.4}$$

2.4.3 Short Time Fourier Transform

The Fourier transforms that were discussed so far hold no information about the times at which frequencies appear. They only indicate that a frequency is present in a signal but not the times at which it appears. However, in a signal that changes over time, this results in information being lost. As an example, the left side of Figure 2.10 shows a signal that changes over time which is given in Equation 2.5. The right side of the Figure shows the Fourier transform of that signal. While it correctly shows the three different frequencies that appear in the signal, there is no information on the times at which they are present.

$$s(t) = \begin{cases} 2 * \sin(2\pi * 4t) + 5 * \sin(2\pi * 2t) & x \leq 10 \\ 3 * \sin(2\pi * 3t) & x > 10 \end{cases} \tag{2.5}$$

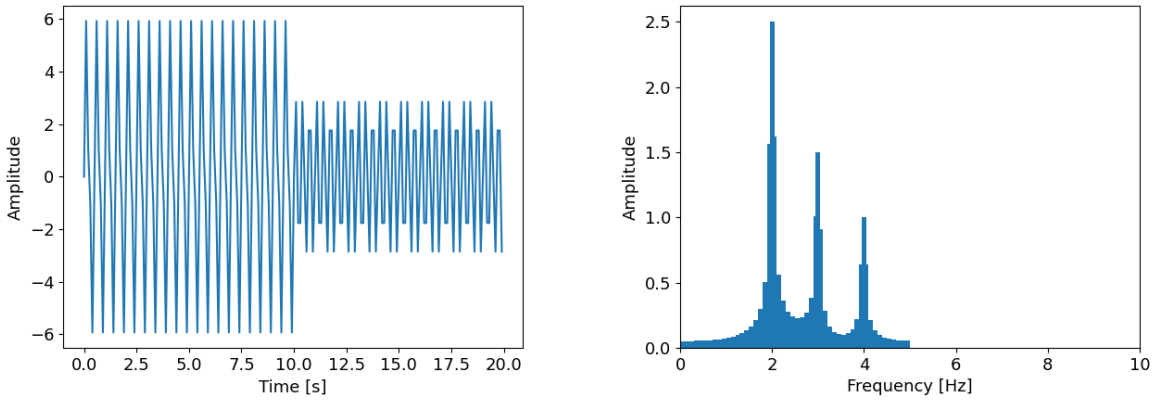

Figure 2.10: Left: A simple changing signal, Right: Fourier transform of the signal

A solution to this problem is the *Short Time Fourier Transform* (STFT). Here, the signal is split into shorter segments for which separate Fourier spectra are calculated. From this information a *spectrogram* can be plotted, which shows the intensity of the frequencies for each time segment. [29]

In practice, this is performed by multiplying a window function which is non-zero for a small interval with the signal. At each step, the window is slid further over the data to analyze every interval separately. Usually the intervals which are calculated are overlapping to reduce artifacts at the edges of the windows. [29]

There are several different window functions. One basic example is the rectangular window, which applies no tapering to the signal segment. While it provides the best frequency resolution, it also leads to the highest spectral leakage due to its abrupt

edges. This makes it suitable only for ideal or theoretical signals where leakage is not a concern. [27]

A more effective window function for practical applications is the Hann window. Its mathematical form is a raised cosine curve, pictured on the left side of Figure 2.11. The Hann window smoothly tapers to zero at both ends, which helps reduce spectral leakage significantly compared to the rectangular window. While it slightly compromises frequency resolution due to a wider main lobe, it strikes a good balance between resolution and leakage. This makes it especially useful in time-frequency analysis tasks such as audio and speech processing. [27]

The right side of Figure 2.11 shows the result of the multiplication of the signal shown in the left side of Figure 2.10 with the Hann window. The resulting signal only shows a small part of the original. If a Fourier transform is performed for every step of the window, this allows for a time resolution in addition to the frequency resolution that is present in the regular Fourier transform.

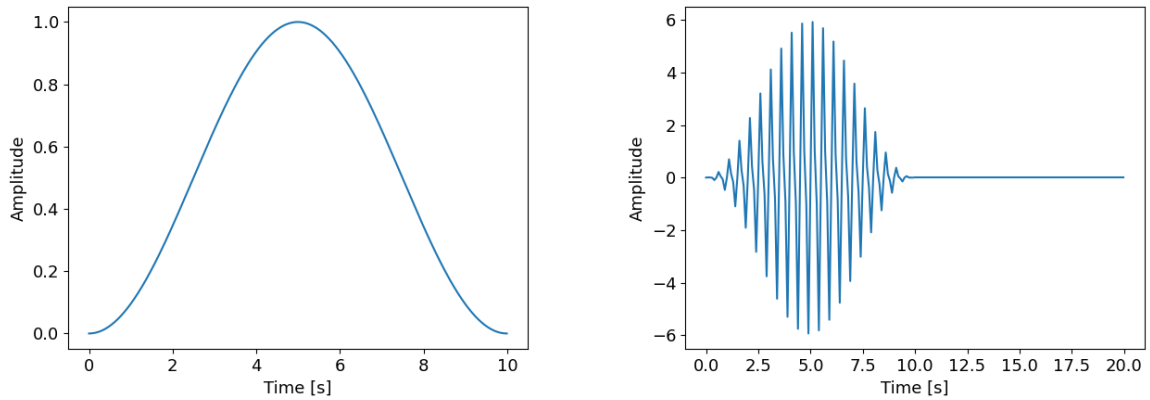

Figure 2.11: Left: Hann window with a length of 10s, Right: Signal of Figure 2.10 (left) multiplied with the Hann window

The result of the STFT is called a spectrogram. Figure 2.12 shows the spectrogram of the signal in the left side of Figure 2.10. It was calculated with a Hann window of 10s with an overlap of 5s. The x-axis shows the time, while the y-axis shows the frequencies that are present at that time. The colorbar gives information on the intensity of the frequencies. The intensity is given as the *power spectral density*, which is the square of the amplitude. Its unit is the decibel, which is defined on a logarithmic scale. [29]

The smallest frequency that can be resolved with the STFT is called the *Rayleigh frequency*, which is defined a $\frac{1}{T}$Hz, where T is the window length. As in the regular Fourier transform, the highest resolvable frequency is the *Nyquist frequency* $f_N = \frac{f_S}{2}$

where f_S is the sampling frequency [27].

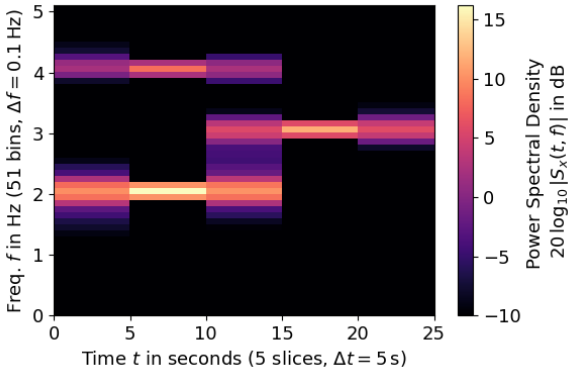

Figure 2.12: spectrogram of the signal in Figure 2.10 (left) created with STFT with a windowsize of 10s and overlap of 5s

The length of the windows has to be chosen carefully, as a shorter window leads to a high time-resolution but a low frequency-resolution. The left side of Figure 2.13 shows the spectrogram of the above signal calculated with a very short Hann window of 2s and an overlap of 1s. On the other hand, a longer window results in a high frequency-resolution but more time data gets lost. This can be observed on the right side of Figure 2.13, where the same signal as in the left side of the Figure was analyzed with a Hann window of 20s and an overlap of 10s. The effect, that a heightened frequency-resolution comes at the cost of a low time-resolution and vice versa is called the *uncertainty principle*. [27]

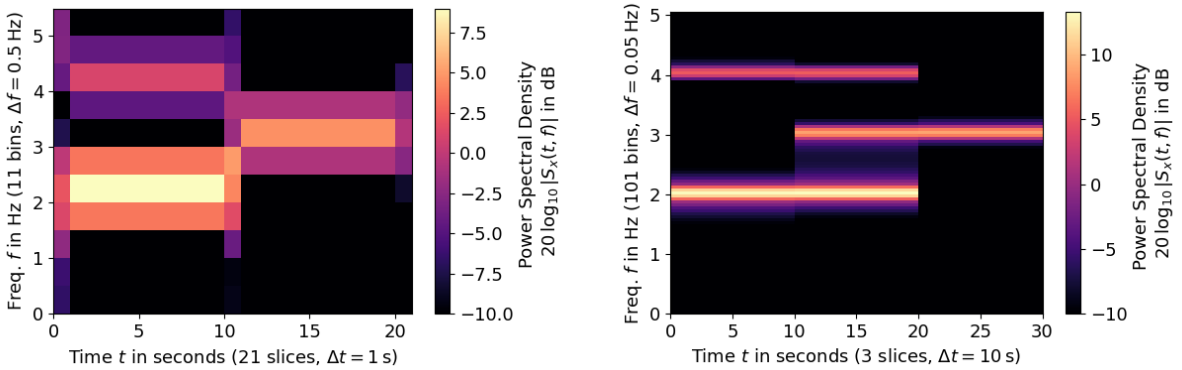

Figure 2.13: Left: spectrogram of the signal in Figure 2.10 (left) calculated with a windowsize of 2s and overlap of 1s, Right: spectrogram of the signal in Figure 2.10 (left) calculated with a windowsize of 20s and overlap of 10s

2.5 Machine Learning for Classification tasks

This thesis aims to apply classification algorithms to recognize vibration patterns that occur before a REPA malfunctions. For the task of classifying data, machine learning, a subclass of artificial intelligence is well suited. Machine learning algorithms are algorithms that can perform tasks without being specifically instructed on the needed steps. They can learn to generalize from training data to make predictions on unseen data [30]. Usually, the training is divided into three approaches: supervised learning, unsupervised learning and reinforcement training.

Supervised learning algorithms are trained with labeled data. The training data is given as input and output pairs from which the model is supposed to learn how to map the input to the output. The goal is to learn a general rule to predict the output. [31]

Unsupervised learning does not require labeled input data. The algorithm is supposed to find structure on its own without any given outputs. Common tasks for unsupervised learning algorithms include clustering and dimensionality reduction. One challenge in unsupervised learning is the difficulty of quantifying the quality of the model outputs. There are also algorithms that employ a mixture of supervised and unsupervised training, where only part of the data is labeled. These are called semi-supervised algorithms. [32][31]

In reinforcement learning, the goal is for the model to learn how to interact with a changing environment. The learner reacts to the environment and receives rewards for every action with the objective of maximizing the reward. A key point of reinforcement learning is the exploration vs exploitation dilemma, where a balance between exploring unknown actions to maximize knowledge and exploiting the already gained knowledge must be found. Examples where reinforcement training is used include automated driving, optimizing energy storage and game engines such as chess engines [33][34][35]. [31]

Supervised learning can be further divided into regression and classification tasks. Regression is used to predict continuous labels. The goal is to find a function that maps the input x to the continuous labels y. Classification is used for data with discrete training labels in order to divide the data into output classes. Algorithms that perform these tasks are also called classifiers. A subclass of classification is the probabilistic classification, where not only the most likely class label is given by the classifier but for every output class a probability is given. This can for example be helpful when

two classes have similar probabilities. This work will only focus on binary classification algorithms which distinguish between two classes. The performance of these models can be quantified by the metrics explained in Chapter 2.5.1.[36]
Popular algorithms used to perform classification tasks include Support Vector Machine, Random Forest, Kernel estimation and Artificial Neural Networks [37][38][39] [40]. The latter will be further explained in Chapter 2.5.2.

2.5.1 Performance Metrics for Binary Classification Models

The performance of classification models can be quantified by several different metrics. The first is the *accuracy*, which measures how well the predicted labels match the labels given by the training set. It is defined by the number of correctly classified objects divided by the total number of objects [41]. While this metric is the easiest to intuitively understand, in unbalanced datasets where one class is underrepresented or classes have different error costs, this metric can be lacking. [42]
A more detailed evaluation of the models' performance can be achieved by calculating the *confusion matrix*. The predictions are divided into true positive (TP), false negative (FN), false positive (FP) and true negative (TN). This is illustrated in Figure 2.14. The relation of these values to the accuracy is defined by Equation 2.6.[42] [41]

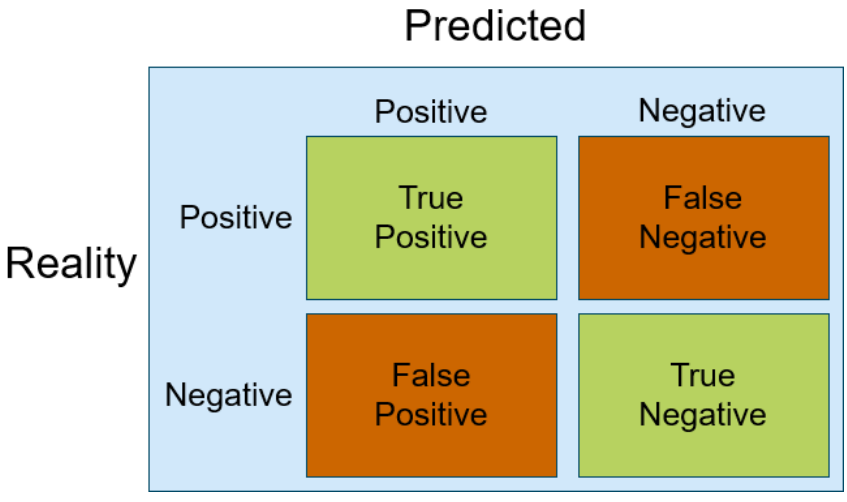

Figure 2.14: Illustration of the confusion matrix

$$\frac{TP + TN}{TP + FP + TN + FN} \tag{2.6}$$

If there is a different cost of false positive and false negative it is beneficial to also calculate the *precision* and *recall* of the model. The former quantifies the amount of correct predictions of the positive class (Equation 2.7) while the latter measures how many of the positive instances in the dataset were found (Equation 2.8) [41]. If false negative is the costlier error, the recall should be maximized, while the precision should be maximized when false positive has the higher cost. A high precision is achieved by a model that is not generating many false positives, however this can also mean, that the model is not generating many positive predictions overall. A model that is always predicting the negative class would have a precision of 1. On the other hand, a high recall means, that the model has few false negatives and can thus correctly identify most positive instances. [42]

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.7}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.8}$$

Precision and recall can be connected into a singular metric by calculating the *F1 score*, which is the harmonic mean of precision and recall (Equation 2.9) [41]. Since it takes into account both false positive and false negative errors it is well suited for imbalanced datasets. If the cost of false negative and false positive are different the F1 score can be weighted by introducing the weight factor β, which represents the factor by which the recall is more relevant than the precision (2.10). This weighted F1 score is sometimes referred to as the *F2 score*. [42]

$$F1 = \frac{2 * (\text{Precision} * \text{Recall})}{\text{Precision} + \text{Recall}} \tag{2.9}$$

$$F1_\beta = (1 + \beta^2) * \frac{\text{Precision} * \text{Recall}}{\beta^2 * \text{Precision} + \text{Recall}} \tag{2.10}$$

2.5.2 Artificial Neural Networks

Artificial Neural Networks (ANN) are modeled after the neuron structure of the human brain. They belong to the deep learning algorithms which are a subset of machine learning algorithms. Similarly to the human brain, ANNs consist of many simple neurons which are connected to form a complex network[43]. In ANNs the biological model of a neuron has to be abstracted as a mathematical function. The first version of the artificial neuron was proposed by Warren McCulloch et al. in 1943 [44].
A neuron consists of a *net value function* ξ and an *activation function* ϕ which are used to transform a set of inputs **x** into an output *o* [45]. Before being processed by the neuron, the inputs are multiplied by a set of weights **w**. The *net value function* then adds the weighted inputs. The result of this function is processed by the *activation function* which limits the amplitude of the neurons output. Before applying the activation function an external bias *b* might be added to the functions input. The neuron *k* can be described by equations 2.11 and 2.12. Equation 2.11 describes the *net value function* with *m* denoting the size of the input vector **x**. Equation 2.12 describes the neurons output which is calculated by the activation function ϕ with added weight *b*. [46]

$$u_k = \sum_{j=0}^{m} w_{kj} * x_j \tag{2.11}$$

$$y_k = \phi(u_k + b_k) \tag{2.12}$$

While a single neuron is mathematically simple, the complexity of an ANN comes from the connection of many neurons. The most common version is the feed-forward network, which consists of several layers of neurons. The output of each layer is used as the input of the next layer, with the exception of the last layer. The output of the last layer is the networks output. Accordingly, this layer is also referred to as the networks output layer. The first layer is also called the input layer, since the networks input serves as the input of all of the layers neurons. The behavior of the layers in between the input and the output layer cannot be directly accessed from outside of the network, which is why they are referred to as hidden layers. If every neuron is connected to every neuron of the next layer, the network is called fully connected. Figure 2.15 shows a simple fully connected artificial neural network with one hidden layer. The amount of hidden layers is described by the depth of the network, which is why this field of machine learning is also called *deep learning*.[45][47]

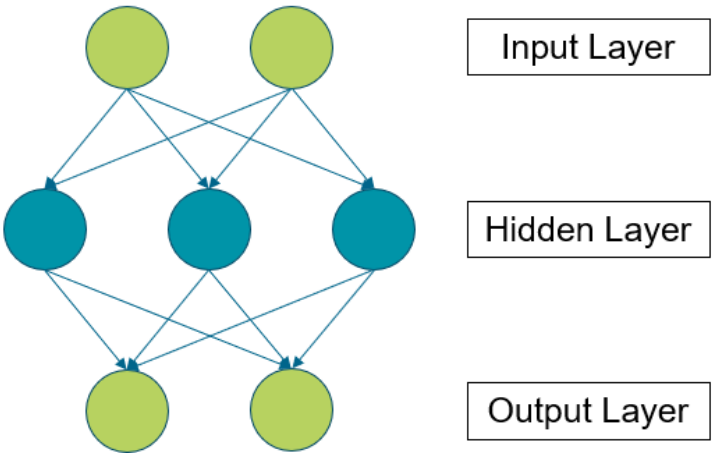

Figure 2.15: Illustration of a fully connected artificial network with one hidden layer

Training of Artificial Neural Networks

In the training process of a neural network, the goal is to teach the network to approximate a function that maps inputs to outputs in a way that minimizes the difference between the predicted and actual outputs. This is done by adapting the weights of the connections between the neurons over time. The networks ability to approximate increasingly complex functions is related to its depth, meaning that deeper networks can model more intricate relationships within the data. The training is usually done by using an algorithm which consists of two phases: the forward phase and the backward phase [47].

In the forward phase, training data is passed into the network, starting at the input layer and propagating through each hidden layer until it reaches the output layer. Each layer in the network applies a mathematical transformation to the input data it receives. The output generated by the final layer is the networks prediction for that particular input. This prediction is then compared to the actual label of the input data. The difference between the predicted output and the true output is called the *loss* of the network. [47]

Once the loss has been calculated, the network enters the *backward phase*, where the goal is to update the weights of the network in such a way that the loss is minimized. This is done by applying a *backpropagation algorithm*. Backpropagation works by computing the gradient of the loss function with respect to each weight in the network. The gradient tells us how much a small change in each weight will affect the overall loss. [48]

The *learning rate* λ controls, how big the update steps of the weights are. The training process is done in multiple epochs. In each epoch, all of the training data is propa-

gated through the network. Usually, the data is passed through the network in batches of several training examples for which the loss is calculated cumulatively. The amount of examples per batch is characterized by the *batch size* and has an important influence on the success of the training. [48]
The success of the networks training is dependent on the loss function, the activation function and the optimizer. In the following sections their most important examples for binary classification will be discussed. There are also several problems that might occur during training, some of which will also be covered in the following.

Loss Functions for Artificial Neural Networks

There are several different possible functions to calculate the loss of the network. The loss function has to be chosen carefully, as it heavily influences the models training.
One loss function that can be used for binary classification, although it is more commonly applied in regression tasks, is the *Mean Squared Error* (MSE). MSE calculates the square of the difference between the predicted probability and the actual label. Equation 2.13 shows, how the MSE loss is calculated, where N is the number of samples, y_i is the true value of the i^{th} sample and p_i is the predicted value of the i^{th} sample [49]. It treats the binary classification problem as a regression problem, where the output is treated as a continuous value. An output of 0 indicates a perfect fit. Since the MSE is a quadratic function is places a larger emphasis on larger errors than smaller ones, which makes it sensitive to outliers. While MSE is simple and differentiable, it is not optimal for classification tasks, as it does not penalize miss-classifications as effectively as other loss functions designed specifically for classification, and it does not capture the probabilistic nature of binary classification as well as other loss functions, which will be discussed in the following paragraphs. [42]

$$L_{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - p_i)^2 \tag{2.13}$$

A much more common and effective loss function for binary classification is *Binary Cross-Entropy Loss* (BCE Loss), also known as Log Loss. BCE is derived from the likelihood of a Bernoulli distribution and measures the dissimilarity between the predicted probability and the actual label [50]. The definition of this loss function is given

in Equation 2.14, where y is the true class label and p is the predicted probability, that the sample belongs to the positive class. This loss function is particularly useful when the model's output is a probability, which is common in binary classification tasks. BCE penalizes the model more when its predicted probability is far from the true class label, making it a natural fit for binary classification problems. For datasets with a class-imbalance, the BCE loss can be modified by adding weights to the samples. If samples of an under-represented class get assigned a higher weight, the model may pay more attention to these samples, balancing out the under-sampling. This modified version of BCE loss is called the *Weighted Binary Cross Entropy Loss* (WBCE). [42]

$$L_{BCE} = -\big(y_i \log(p_i) + (1 - y_i) \log(1 - p_i)\big) \tag{2.14}$$

Another loss function that can be used in binary classification, although less common in deep learning, is *Hinge Loss*, which is typically associated with Support Vector Machines [51]. The calculation of Hinge loss is given in Equation 2.15, where y is the true label of the instance and $f(x)$ is the predicted output for x. Hinge loss is used in margin-based approaches, where the goal is to maximize the margin between classes. It encourages the model to output a value that is further away from the decision boundary, ensuring that positive examples are classified as +1 and negative examples as -1 with a margin of at least 1. Though not as widely used in deep learning as Binary Cross-Entropy, Hinge Loss can still be employed when margin-based learning is the focus. [42]

$$L_{Hinge} = \max(0, 1 - y_i * f(x_i)) \tag{2.15}$$

For situations where a dataset is heavily imbalanced, *Focal Loss* is a modification of Binary Cross-Entropy that can help the model focus more on the hard-to-classify examples [52]. It is defined by Equation 2.16, where p_t is the predicted probability, that the sample belongs to the true class, α_t is a weighing factor that controls the importance of the sample and γ is a focusing parameter that controls, how fast the easy samples are down-weighted. Focal Loss down-weights the loss for well-classified examples and places more emphasis on misclassified or harder examples. It is particularly useful for imbalanced datasets, where the model may otherwise bias its predictions toward the

majority class. [42]

$$L_{Focal} = -\alpha(1 - p_t)^{\gamma} * \log(p_t) \tag{2.16}$$

Another useful loss function, especially in cases where the model outputs a probability distribution over the classes, is the *Kullback-Leibler Divergence* (KL Divergence) [53]. The definition of the KL divergence is given in Equation 2.17, where $p(x_i)$ is the true probability of the class x_i and $q(x_i)$ is the predicted probability of class x_i. KL Divergence measures how one probability distribution diverges from another, and it can be used in binary classification when the model predicts a distribution rather than a single probability. KL Divergence is useful when optimizing a model for probabilistic outputs and ensuring that the predicted distribution closely matches the true distribution of the labels. [42]

$$KL(p||q) = \sum_i p(x_i) * \log\left(\frac{p(x_i)}{q(x_i)}\right) \tag{2.17}$$

The quantification of the models performance after training can be done by *cross-validation*. In this technique, the set of training data is split into two subsets. The first is used to train the model while the second is used as a testing dataset. After the training subset is passed through the network, estimations are done on the testing subset. This way, the testing is done on data the network has not seen before, which makes the validation more accurate. [54]

Problems in the Training of Artificial Neural Networks

There are several problems that may occur in the training process that lead to a bad performance of the network. In the following paragraphs, overfitting, underfitting, vanishing or exploding gradients and class imbalance will be discussed.
Overfitting occurs when a machine learning model learns not only the underlying patterns in the training data but also the noise and details specific to that data. This results

in a model that performs well on the training set but poorly on unseen data, as it fails to generalize to new, unseen examples. The model becomes too complex and captures irrelevant information, leading to poor generalization. [55] Underfitting on the other hand occurs, when the model is too simple to capture the patterns in the data. The model fails to generalize well because it does not adequately represent the relationships in the data.

To overcome overfitting, techniques such as dropout and early stopping can be employed to prevent the model from becoming too complex. Cross-validation can also be used to ensure the model performs well on unseen data. Dropout prevents overfitting by randomly setting a fraction of the neurons' activations to zero during training [56]. This forces the model to rely on different subsets of neurons in each training iteration, promoting more robust learning. By preventing the model from becoming overly reliant on specific neurons, dropout encourages generalization and improves performance on unseen data. When using early stopping, training is stopped when the models performance on a validation set stops improving, preventing overfitting [57]. This helps avoid training the model too long, allowing it to generalize better by stopping at the point where it has learned enough without memorizing the training data. Additionally, the chosen batch size during training can impact overfitting. A small batch size reduces overfitting by introducing more noise into the gradient updates during training. This noise helps prevent the model from memorizing the training data too precisely, allowing it to explore a broader range of solutions. As a result, the model is less likely to overfit to the specific patterns or noise in the training set, leading to better generalization on unseen data. [48]

Underfitting can be solved by using a more complex network or switching to a different network architecture or introducing more layers. However, this has to be done carefully because a network that is too complex might result in overfitting. [48]

Vanishing gradients happen when the gradients of a neural network become very small during backpropagation, making it difficult for the model to learn, especially in deep networks. This is due to the gradients shrinking as they are propagated back through the layers, effectively preventing weight updates in earlier layers. Exploding gradients, on the other hand, happen when the gradients become excessively large, causing instability in the learning process and potentially leading to overflow or divergence in the models weights. [43]

One possibility to avoid vanishing or exploding gradients is proper weight initialization,

which helps prevent gradients from vanishing or exploding by setting weights appropriately. This means, that the initial values of the networks weights have to be set in a way that ensures stable gradient flow during training. Methods like He initialization or Xavier initialization adjust the weights based on the number of input and output units in each layer, preventing gradients from vanishing or exploding. The weight initialization has to be chosen in accordance to the used activation function. Gradient clipping can also be applied to limit the size of gradients during backpropagation, preventing them from becoming too large. Here, the gradients are scaled when they reach a certain threshold. This ensures, that the gradients during backpropagation remain within a manageable range, improving model stability, especially in deep networks or when training with long sequences. Batch normalization further stabilizes training by normalizing the output of each layer, ensuring a consistent distribution of activations and reducing the risk of gradient issues. Finally, employing smaller learning rates helps control gradient updates, preventing large steps that could cause exploding gradients. [43][48]

Class imbalance occurs when the classes in a dataset are not represented equally, with one class having significantly more examples than the others. This imbalance can lead to biased model training, where the model tends to predict the majority class more often, neglecting the minority class. As a result, the model may have poor performance on the underrepresented class, which can be particularly problematic in tasks like fraud detection or medical diagnosis, where the minority class is often the more important one to predict accurately. One approach to solve this is resampling, where you either oversample the minority class or undersample the majority class to balance the dataset. Another method is to use class weights during training, which gives more importance to the minority class by assigning a higher weight to it in the loss function. Additionally, data augmentation can help increase the size of the minority class, and anomaly detection techniques can be applied when the imbalance is extreme. Finally, ensemble methods like random forests or boosting can also help by incorporating multiple models to improve performance on imbalanced data. [48]

These challenges in the training of ANNs can also be adressed by adjusting the activation function and the optimizer. In the following sections, common activation functions and optimizers will be introduced.

Activation Functions for Artificial Neural Networks

For a successful training, the correct choice of activation functions is important.There are several different activation functions that can be used in a neuron, including the *Rectified Linear Unit Function* (ReLU), *hyperbolic tangent* and the *sigmoid* function. All activation functions have in common, that they map the input on a limited output space.

The ReLU-function, which can be seen in Equation 2.18, equals the identity function for positive values and zero for negative values. The linear behavior of the function offers many advantages for training. While the first order derivatives are large and consistent for positive values, the second derivative is zero. The absence of second-order effects makes the gradient direction more useful for training. The activation function also allows for sparse activation, as only 50% of the hidden units are activated in a randomly initialized network. However, since the outputs are not centered around zero and always non-negative, the learning in backpropagation might be harder. This problem can be addressed by applying batch normalization. ReLU can also lead to dying neurons, where they become inactive and output zero for almost all inputs. When this happens, no gradients flow backwards during backpropagation and the neuron dies. [48]

Several different variants of ReLU address the dying neurons problem, for example *leaky ReLu*, which applies a positive gradient α smaller than one for values lower than zero, commonly between 0.01 and 0.3 [58]. Another variant is *parametric ReLU*, where the gradient α is learnable by the model [59]. [48]

If a normalization of the model output is needed, using the sigmoid activation function (Equation 2.20) might be reasonable, as this function maps the input to the interval $[0, 1]$. However, since the function is only highly sensitive to a change in the input if the input is close to 0 and has otherwise very small gradients, gradient-based training can be very difficult.

While the hyperbolic tangent function is similar to the sigmoid function, it can be better suited for training. This is due to the function being centered around zero and not around $\frac{1}{2}$ as the sigmoid function. This leads to a higher similarity to the identity function for inputs close to zero. [48]

$$f(x) = max(0, x) \tag{2.18}$$

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.19}$$

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.20}$$

Optimizers for Artificial Neural Networks

Another important factor in the training process is the optimizer choice. The optimizers task is to adjust the models weights to minimize the loss function. They use the gradients computed during backpropagation to update the weights in a way that improves the models predictions. Popular optimizers like *Stochastic Gradient Descent* (SGD), *Momentum*, *Adaptive Moment Estimation* (Adam), and *Root Mean Square Propagation* (RMSprop) employ different strategies to adjust the weights and handle challenges like vanishing gradients or slow convergence.[60]
The SGD optimizer is an optimization algorithm that updates the models weights based on the gradient of the loss function with respect to the weights [40]. Unlike traditional batch gradient descent, which computes the gradient using the entire dataset, SGD updates the weights after processing each individual training example, making it more computationally efficient for large datasets. While this leads to noisier updates, it can help escape local minima and converge faster, especially when combined with techniques like momentum or learning rate schedules. [61]
If there are areas in the loss landscape, where the surface curves much more steeply in one dimension than in the other, SGD can not be the optimal choice. These areas often occur close to local minima. The Momentum optimizer addresses this weakness by a fraction of the update vector of the past time step to the current update vector. Thsi results in faster convergence and a reduced oscillation. [60] RMSprop is an adaptive learning rate optimizer that adjusts the learning rate for each parameter based on the average of recent squared gradients. It helps to address the issue of learning rates being too large or too small by dividing the learning rate by a moving average of the squared gradients, which smooths out oscillations in the parameter updates. This optimizer is particularly effective for training models on non-stationary objectives, such

as recurrent neural networks, and is known for its stability in scenarios with noisy or sparse gradients. [60] The Adam optimizer combines the advantages of the optimizers Momentum and RMSProp to improve the efficiency and performance of the learning process. It works by maintaining two moving averages for each parameter during training. The first moment is the exponentially weighted average of the gradients, while the second moment is the exponentially weighted average of the squared gradients. These averages help adapt the learning rate for each parameter, allowing the algorithm to update weights more effectively and with greater stability. One of Adam's key features is its ability to adjust the learning rate for each parameter individually, which helps the model converge faster and reduces the need for manual tuning of learning rates. Additionally, Adam includes bias correction to account for the initial biases in the moving averages, particularly in the early stages of training when the averages are still being initialized. [62]

2.5.3 Convolutional Neural Networks

For the task of handwritten letter recognition, LeCun et al. proposed a neural network containing *convolutional layers* and *pooling layers* in 1989 [63] called *convolutional neural networks* (CNN). They are well suited to process data in the form of multiple arrays, such as 2D arrays for images or spectrograms created by STFT. CNNs work by exploiting the fact, that in array data values are usually highly correlated to neighboring values, forming so-called *motifs*. Motifs are made up of distinct *features* which can vary in their relative position to each other. A CNN trained to recognize animals for example might learn that the motif *cat* contains the features *tail*, *head* and *paws* which can appear in different relative positions. The CNN detects the motif by coarse-graining the position of each feature, which makes the recogniton invariant to shifts in the relative position. CNNs are generally made up of *convolutional layers*, *pooling layers* and *fully connected layers*. [64]

The working principle of a convolutional layer is illustrated in Figure 2.16. An example of a two-dimensional input array can be seen in the middle of the Figure. To reduce the spacial dimensions of the input, a matrix, called *kernel* is slid over the input. A simple example kernel is shown on the left side of the Figure. At each position, the

dot product of the kernel and the covered area is calculated as seen on the right side of the Figure. The output array contains all multiplication results. During training, the model finds an optimal kernel. The convolution operation is defined by the size of the kernel, in the case of the example this is four, and the *stride*. The stride determines how many pixels further the kernel is shifted at each step. The higher the stride, the smaller the overlap is between the convolution operations and thus the smaller the output. To reduce border effects a padding can be added around the input. This prevents border effects, as otherwise the data on the edges and corners would have less impact than the data in the middle of the input which appears in more convolution operations. In the Figure the padding is shown in gray. There are several different padding techniques available, such as *zero padding*, which is shown in the picture. The padding in the picture has the size one, since there is one layer of zeros added around the input. In praxis there can also be added more layers, especially if a large kernel size is used. The output of a convolutional layer is called *feature map*. Since each neuron in the output is only connected to a small region of neurons in the input, the convolutional layer focusses on local patterns. The *receptive field* of a neuron quantifies the size of the neurons input. [63] [65]

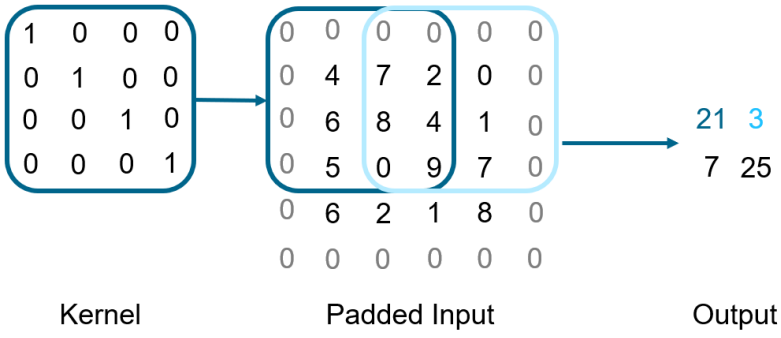

Figure 2.16: Illustration of a convolutional layer

In each layer, the convolution operation is repeated with multiple kernels, each optimized to detect a specific feature. The output of the layer consists of all feature maps that are produced by the convolution operations. The different maps are also called the *channels* of the output. A colored image for example consists of three channels, each channel giving a RGB value for a single pixel.

The spatial dimensions of the output of a convolutional layer are dependent on the input size D_{in}, the kernel size K, the padding P and the stride of the kernel S. The relationship between these parameters is shown in Equation 2.21.

$$D_{out} = \frac{D_{in} - K + 2P}{S} + 1 \tag{2.21}$$

Convolutional layers are usually followed by pooling layers, which further reduce the spacial dimensions of the input. Figure 2.17 shows an illustration of the working principle of a pooling layer. The input is divided into smaller regions, which are then clustered into a single value. *Max pooling*, which is shown in the Figure, uses the highest value that can be found in the cluster. Another popular pooling strategy is called *average pooling*, here the average value in the cluster is calculated. [65]

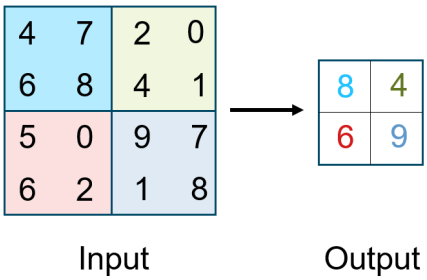

Figure 2.17: Illustration of max pooling

In most CNNs used for classification tasks, the convolutional and pooling layers are followed by one or multiple fully connected layers. The amount of neurons in the first fully connected layer usually equals the number of channels after the last convolutional layer. Each neuron is connected to each neuron of the previous layer. [64]

3 Experimental Setup

In order to develop a predictive maintenance strategy for the REPAs, first the entire life cycle needs to be understood. Since in a regular PTC that performs on rotation cycle per day, a life cycle takes about 25 years, an accelerated test has to be performed. For this, the REPA test rig was designed [66].
In this Chapter, the general assembly of the test rig will be explained, followed by a detailed description of the installation of the vibration sensors that collect the data analyzed in this thesis.

3.1 REPA Test Rig

The REPA test rig, which is located at the Plataforma Solar de Almería in Spain, was designed to analyze the life cycle of REPAs in a shorter time span. The test rig can simulate up to 570 rotation cycles per day, which means that the estimated lifespan of 10,000 cycles can be achieved in under 20 days. Two REPAs are tested simultaneously, which is compareable to a real PTC where there is a REPA on each side of the Collector. [66]
To provide conditions that are as similar as possible to a real PTC, the REPAs are rotated and moved translationally at the same time, with the translational movement simulating the expansion of the receiver tubes. Additionally, the REPAs are part of a HTF cycle providing hot fluid of 390°C and 30bar.[66]
Figure 3.1 shows an overview over the REPA test rig. The test rig consists of three main components: the main assembly, the HTF cylce and the supervisory control and data acquisition system (SCADA). The former contains the kinematics unit, which performs the rotational and translational movements to simulate the daily movements occurring in a PTC. The HTF cycle is responsible for keeping the HTF at the correct temperature, pressure and mass flow. Lastly, the SCADA system is used to collect data and to monitor and control the test rig. In the following sections, these components will be explained in detail.

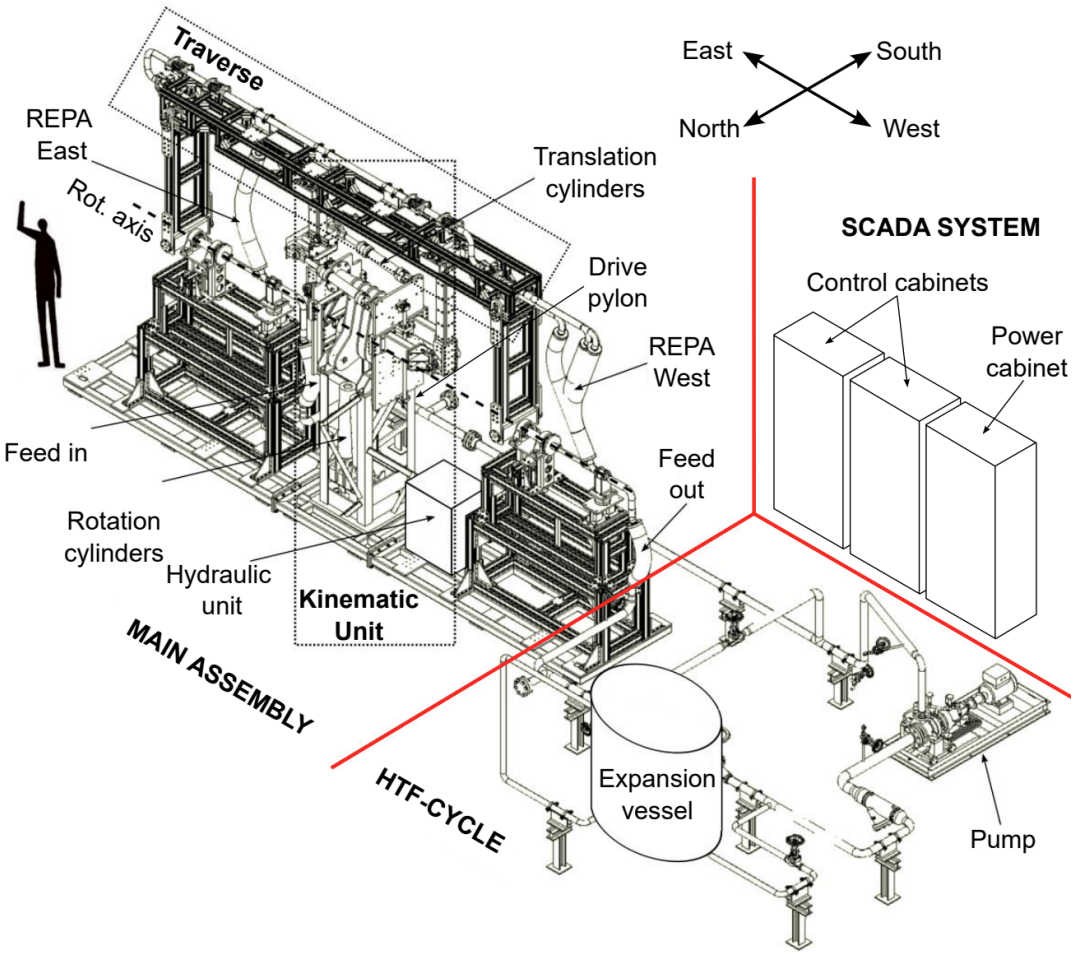

Figure 3.1: Assembly of the REPA test rig [67]

3.1.1 Main Assembly

The main assembly of the REPA test rig includes the kinematic unit, the traverse and two fixtures for the REPAs. Figure 3.2 shows the components of the main assembly with RFHAs installed as REPAs.

The kinematic unit is a hydraulic system consisting of a hydraulic unit and four hydraulic cylinders. It also includes a drive pylon. Two of the cylinders, which are located in the center of the drive pylon, generate the rotational movement, while the other two cylinders are placed on top of the drive pylon and perform the translational movement. The traverse is connected to the kinematic unit by two drive pylon arms.

The two REPAs are connected on their upper ends by the traverse, which simulates the thermal expansion of the receiver tube in a real PTC. The traverse movement is executed in the east-west direction. The motion is performed by the two hydraulic translational cylinders which rotate the drive pylon. Thus, the translational motion is actually a rotation which is measured by the translation angle θ. Since only 500-600mm of heat

dilation have to be simulated, only small values are chosen for θ, meaning that the small angle approximation is valid in this case. Therefore, a linear relation between θ and the absorber pipe temperature can be assumed. The maximum translation angle in the west direction is $\theta_1 = -5°$, while in the east direction it is $\theta_2 = 12°$ (see Figure 3.2). This corresponds to a translational movement of 500mm. [66]
The hydraulic cylinders that are responsible for the rotational movement move the traverse around the rotational axis, which is denoted in red in the Figure. It allows for a rotation from -23°, which is the stow position, to the end position of 186°. In the Figure these are the angles α_5 and α_4. The start angle, which would be the angle that the PTC has at sunrise, is α_3. In the REPA test rig, the rotation is performed in a fast, continuous motion as opposed to a real PTC where the movement from the start angle to the end angle is slow and incremental.
On both sides of the traverse a REPA is attached. When the traverse is moved, the movement translates to a movement of the REPAs. The other ends of the REPAs are welded to the feed-in and feed-out pipes of the HTF cycle. In the first test campaign that is analyzed in this work, RFHAs are used as the REPAs. It was completed before this work begun. The swivel joint is attached to the table of the main assembly. The torque sword is welded directly to the pipe segment extending from the swivel joint. Both components are aligned along the test rigs rotational axis. To reduce heat loss, both the corrugated flex hose and the swivel joint are insulated. The left side of Figure 3.3 shows the installed RFHA with insulation.
The second test campaign examines BJAs. It consists of three individual ball joints with pipe connections between them. The translational angle is influenced by the length of the connection between the traverse and the upper ball joint. The highest forces appear in the lowest ball joint, making it the critical point where failure occurs first. In Figure 3.3 the BJA in the test rig is shown without insulation. [67]

3.1.2 Heat Transfer Fluid Cycle

The HTF cycle is the second component of the REPA test rig. It is a pipe system consisting of a pump, an expansion vessel and ten electric heaters. The pump controls the mass flow rate $\dot{m}_{HTF}$ and the pressure p_{HTF}. The former can range from $6\frac{\mathrm{m}^3}{h}$ to $60\frac{\mathrm{m}^3}{h}$ while the latter can be up to 40bar. In case of failure, the pump can be separated

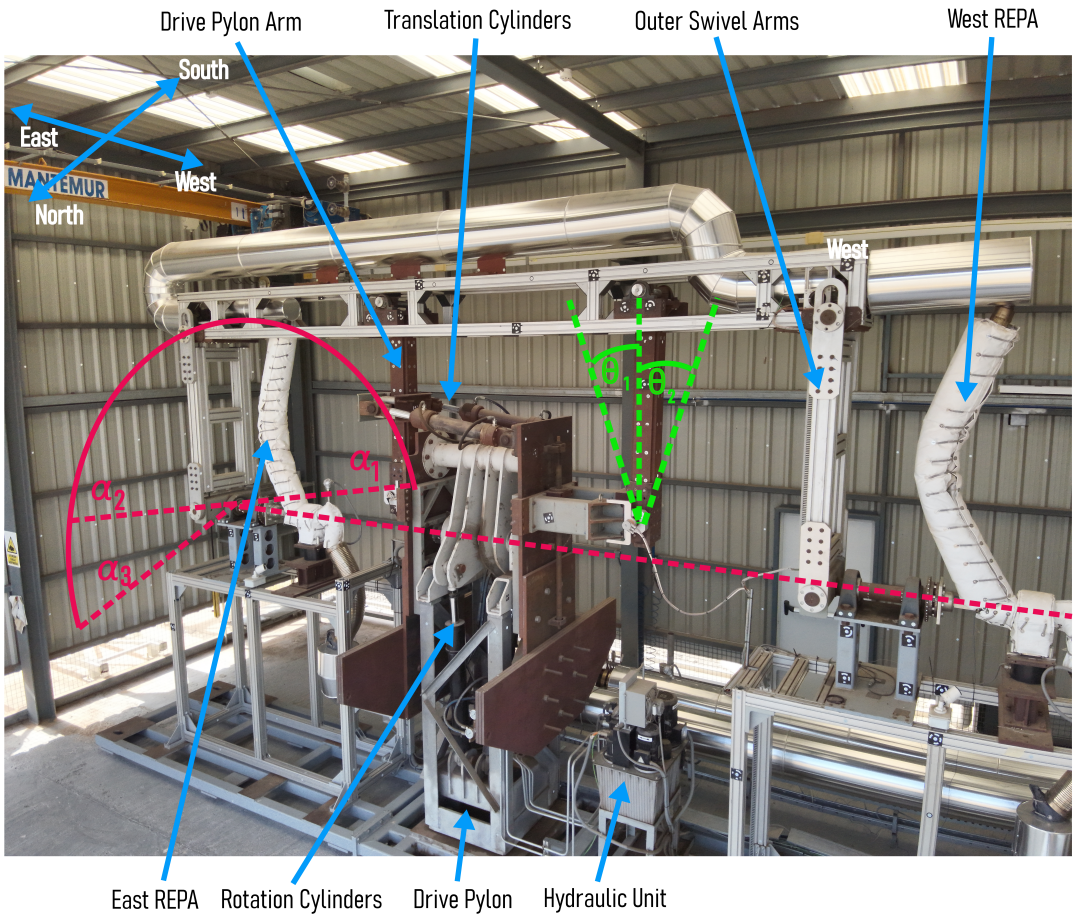

Figure 3.2: Main Assembly of the REPA test rig featuring the kinematic unit, the traverse and two RFHAs, Red: Rotational axis (α_1: start angle, α_2: end angle, α_3: stow angle), Green: Transversal axis (θ_1: cold angle, θ_2: hot angle), adapted from [67]

from the main assembly, ensuring the safety of the system. For this, a bypass around the main assembly is installed in the pipe system. [67][66]

Since the HTF expands when it is heated up, the expansion has to be compensated to regulate the pressure. For this, an expansion vessel is connected to the pipe system. It works by pumping or evacuating nitrogen into the vessel, making it possible to adjust the pressure from 1 to 40 bar.

The temperature of the HTF is provided by ten high-performance electric band heaters. Each heater has an output of 3500W. They are installed on the pipe directly before the feed-in leading to the main assembly. For minimum heat loss, all pipes are surrounded by a insulation with a thickness of 120mm. The maximum temperature that can be achieved by this setup is 450 °C.

The HTF that was used in the test campaigns analyzed in this work is *Therminol ®VP-1* oil. It has a maximum permissible temperature of 400 °C.

Figure 3.3: Left: RFHA with insulation in REPA test rig, Right: BJA in REPA test rig

3.1.3 SCADA system

The SCADA system is responsible for monitoring and controlling the entire test rig, which includes both the kinematic unit and the HTF cycle. Central to this setup is the Siemens SIMATIC S7-300 Programmable Logic Controller (PLC), which links all sensors and actuators through various interfaces, either analog or digital. This PLC regulates the HTF pump speed, collects measurement data, and sends control commands to the servo controller. The servo controller is responsible for managing the hydraulic system's motor speeds and controlling the motion of the cycles.

A crucial part of the system is the Open Platform Communications (OPC) server, which allows for the reading and writing of PLC variables. It connects to a Graphical User Interface built in LabVIEW, enabling remote control of the test rig. Cameras linked via Ethernet also allow for visual monitoring. The servo drive handles the system's ro-

tational movement, while an inverter drive controls the translational motion. The 4/3 directional control valves manage the movement direction of the traverse system, determining whether the cylinder pistons are extended or retracted. The traverse speed slows down at the changeover points near 35° and 133° when a piston shifts from pushing to pulling, or vice versa.
In December of 2024, before the second test campaign was carried out, the existing PLC system was upgraded to incorporate new load cells, and this extension also allows for the addition of more sensors, such as extra load cells or cameras, to support the monitoring of REPA test campaigns. The second cabinet connects to the PLC system via a module connector located in the center of the cabinet. The power supply unit, a SITOP PSU100S, is installed in the top section of the cabinet, while the two new load cells are connected at the bottom. These individual analog inputs are then incorporated into the existing PLC system. [67]

3.2 Vibration Measurements

The data that is analyzed in this work is collected by two uniaxial accelerometers, one on each REPA of the test rig. The vibration sensors are of the model *PCB-(M)320C52* by *PCB Piezotronics*. They allow for measurements with a maximum sample rate of 10,000Hz and a sensitivity of $1.02\frac{\mathrm{mV}}{\mathrm{m/s^2}}$. They can endure temperatures of up to 163℃. The power supply is realized by USB connection to the measurement laptop, which also transfers the collected data to the laptop. [68]
The first analyzed test campaign was performed before this thesis was started. The setup of the vibration data acquisition of this campaign is outlined in Chapter 3.2.1. The second test campaign, which featured ball joint assemblies, was set up and performed in 2025. The installation of the vibration sensors was part of this thesis.

3.2.1 Rotary Flex Hose Assembly Test Campaign

For the RFHA test campaign, the sensors were installed on the bottom of the metal flex hose, in between the the torque sword and the swivel joint. There, the sensor is close enough to the swivel joint to pick up the vibrations occurring there. The gap in the

isolation in this spot provides the space needed to screw the sensor onto the assembly. Figure 3.4 shows the west REPA with the vibration sensor installed.

Figure 3.4: RFHA with installed vibration sensor (marked with red square)

3.2.2 Ball Joint Test Campaign

Since the lowest ball joint is usually the one where failure occurs, they are chosen for the installation of the vibration sensors in the ball joint test campaign. To avoid overheating of the sensors, they are installed on an attachment. The attachment is made of aluminum and long enough to go through the isolation that is placed around the ball joint. It also features three grooves to improve the heat exchange with the environment. The attachment is fastened to the screw of the ball joint. It has a hole at the end, where the vibration sensor can be screwed into. Figure 3.5 shows the lowest ball joint of the west REPA with the attachment for the vibration sensor with and without the isolation.

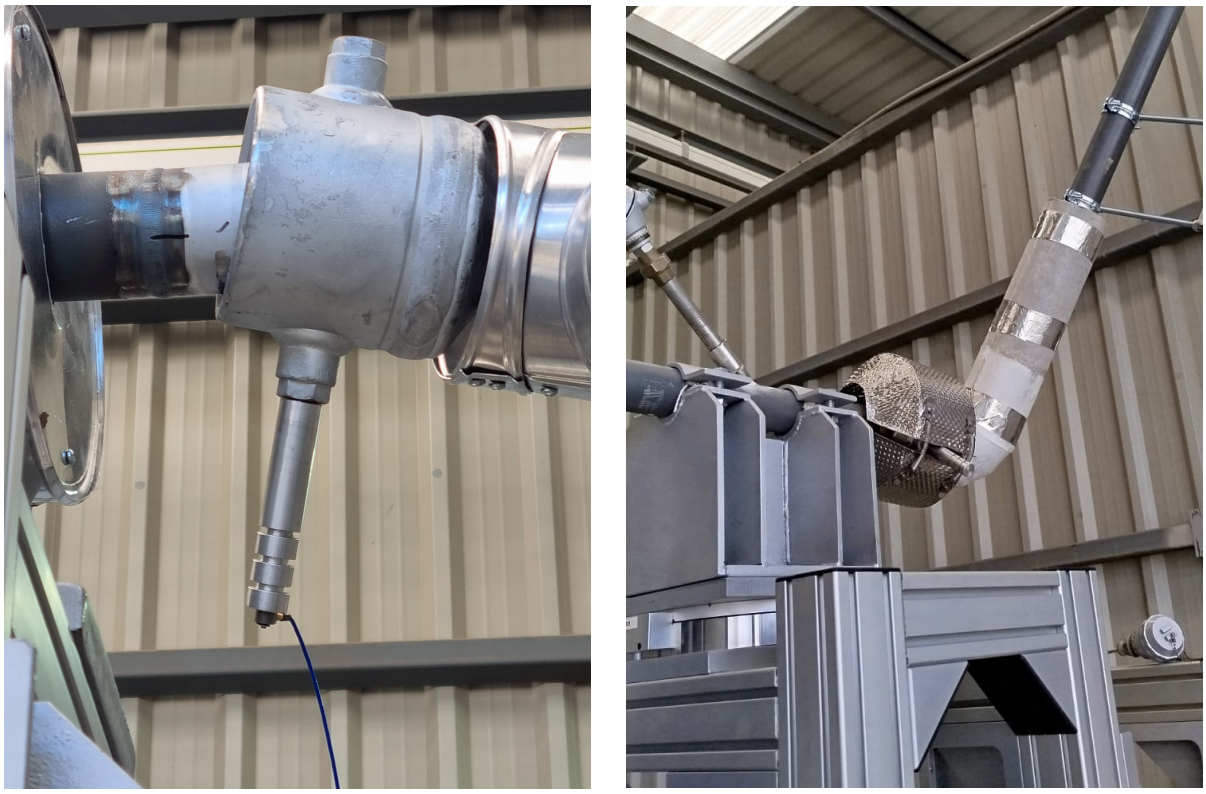

Figure 3.5: Left: Lowest ball joint with attachment for the vibration sensor and the vibration sensor installed, Right: Lowest ball joint with attachment for the vibration sensor and isolation installed

4 Implementation of the computational Methods

This work has the objective to find patterns in the vibrations that occur in the REPAs of the test rig indicating future failures. This enables predictive maintenance of the flexible pipe connectors, which could prevent fires that occur due to leakage of the hot HTF and save cost due to unpredicted down-times. To find patterns, the raw data first needs to be transferred to a format that is suitable for a machine learning model. This process is described in Chapter 4.1. For the recognition of patterns indicating problems in the REPAs, a machine learning model is trained, which is discussed in Chapter 4.2. Python is used as the programming language for the entire thesis.

4.1 Data Preparation

Figure 4.1 shows an overview over the steps of the data preparation process. If not stated otherwise, the calculations described in this Chapter are performed using the *SciPy* library which offers useful functions for data analytics [69]. The handling and processing of the data is done using the Python packages $Pandas$ and $NumPy$ [70][71][72].

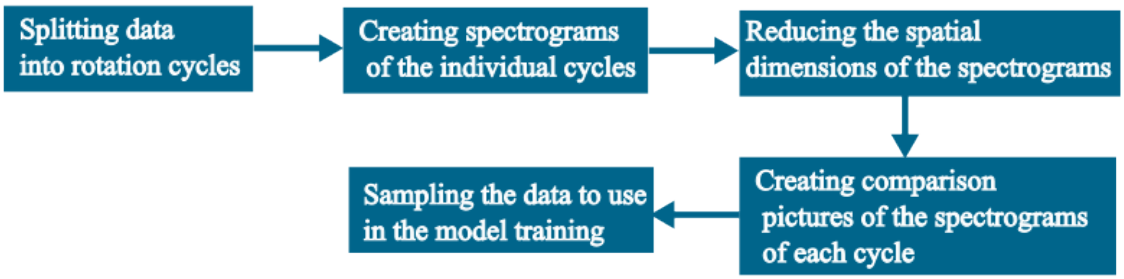

Figure 4.1: Overview of the data preparation process

Firstly, the raw data collected by the vibration sensors needs to be split into the individual rotation cycles that the REPA test rig performs. The rotation data is collected by the SCADA system and saved in a csv file. The system saves the current rotation angle of the test rig every five seconds. Thus, the start and end of each cycle can only

be determined with an accuracy of five seconds. A rotation cycle is defined to start when the rotation angle leaves the minimum value and to end when the angle reaches the minimum value again. In between, the rotation angle has to reach the maximum angle for the cycle to be counted. Since the minimum and maximum angles might not be measured due to the measuring rate of 5s, a margin is added to these values in which the values still count as reached. Some of the cycles are completed at lower temperatures. These are not part of the analysis, since the lower temperature might impact the vibrations.

Once the timestamps of the start and end of each rotation cycle are found, the vibration data recorded during each cycle is analyzed to create spectrograms of the cycles. For this, a Wiener filter is first applied to the data. This is done, because the data contains very high frequencies with high intensities that obscure the underlying patterns. Since the measurement is performed at a sampling rate of 10,000Hz the STFT can only detect frequencies below 5000Hz, with the frequencies above this threshold being aliased as lower frequencies. The Wiener filter flattens the data, removing these very high frequencies and making the underlying patterns visible.

Afterwards, the spectrograms of the cycles are calculated using the STFT-function implemented in SciPy. Since there is a vibration sensor on each of the two REPAs, there are two spectrograms for each rotation cycle. The chosen windowlength for the STFT is 1s. Since the sampling rate equals 10,000Hz, each window contains 10,000 measuring points. The overlap of the windows is half the window size, resulting in a time resolution of 0.5s and a frequency resolution of 1Hz.

Since the cycles differ in their length, their spacial dimensions are reduced to create data of equal sizes. By using average pooling, the sizes of all spectrograms are set to 256x256. The kernel size of the average pooling filter is calculated dynamically based on the original length of the cycles.

For the recognition of failures that will occur soon in the REPAs, the variations that naturally occur in the vibrations due to environmental factors such as wind or different temperatures need to be differentiated from the variations that occur due to malfunction of the pipe connectors. There are two REPAs per PTC, one on each side. Both REPAs are exposed to similar conditions since they are close to each other and perform their movements simultaneously. Thus, it is likely, that in both REPAs similar vibrations occur as long as both function regularly. A malfunction in one of the REPAs would result in a change of the vibrations that can not be explained by environmental factors and

thus does not occur in the other REPA.

Following this thought process, the spectrograms of the vibrations occurring in each of the REPAs are compared for every cycle. To quantify the similarity of the spectrograms, the structural similarity index metric (SSIM) is used, which was proposed by Wang et al. in 2004 [73]. It is used to score the similarity of two images on a scale from 0 to 1, where 0 is given to two images that are completely different and 1 is given to identical images. It is calculated by comparing statistical properties of small windows of the images to each other. The SSIM of the small windows x and y is calculated by using Equation 4.1, where $\mu_{x/y}$ is the pixel sample mean of the windows, $\sigma_{x/y}$ is the pixel sample variance and σ_{xy} is the pixel sample covariance. c_1 and c_2 are used to stabilize the division and are calculated from the dynamic range of the pixel values. [73]

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{4.1}$$

After the SSIM is calculated for every window of the spectrograms, the mean of the values is calculated, resulting in the SSIM of the entire images. Additionally, a new image can be created from the scores of each window, showing the areas with high similarity between the two spectrograms. This is calculated for every cycle of the test campaigns. The calculation is done using the image processing library *scikit-image* [74]. The comparison images are used later to train the neural network.

Since there are more cycles performed where both REPAs function normally than where a REPA malfunctions, class imbalance needs to be avoided. Firstly, under-sampling is used to create classes with an equal amount of samples. Thus, the model is not trained on every cycle with REPAs that function normally but only a selection. Additionally, the loss function can be adjusted in the training to put a bigger weight on the second class, which indicates a failure occurring soon.

4.2 Failure Prediction with Machine Learning

To detect patterns in the vibration data that indicate a REPA malfunction, a Convolutional Neural Network (CNN) is trained. As explained in Chapter 4.1, the CNN is not trained on the actual vibration data but on the comparison pictures of the spectrograms of the vibration data recorded by each sensor. This has the advantage, that variations

that are caused by the environment and not by a REPA malfunction do not appear in the final data, since they influence both REPAs at the same time. Furthermore, this method allows for a more generalized failure prediction. Since for each type of REPA there is only one test campaign available as training data, it is not possible to identify which patterns in the vibrations appear before every failure and which are specific to this test campaign. By only giving the CNN the information on the similarities and differences between the data recorded by the two sensors, the specific patterns are obscured to allow for a more general training. By not using the SSIM score, but the comparison images of the spectrograms, the model can also analyze where in the spectrogram the differences or similarities occur.
Another measure taken to avoid local variations impacting the performance of the CNN is to not use singular cycles as training data but to stack the comparison pictures of several cycles that happen after each other. The chosen amount of stacked cycles is 64. This enables an evaluation of the vibrations that occurred in roughly two months in a real PTC. Thus, the CNN is trained on 3D data of the shape $(64, 256, 256)$. The input of the CNN has the shape $(n, 1, 64, 256, 256)$, where n is the amount of samples that are presented to the network simultaneously, known as the batch size. The second dimension is the amount of channels, in this case one.
The CNN is used for binary classification, where 0 means, that in the analyzed cycles the REPAs function normally, while 1 indicates a malfunction of one of the REPAs occurring soon.
The implementation of the CNN is done in PyTorch, a Python library for building AI models [75]. In the course of this work, different CNN architectures were tested. This chapter only contains the details of the architecture that proved to be the most successful. Table 4.1 gives an overview of the layers that the CNN consists of. There are four convolution layers, each followed by a pooling layer. After the last convolution an additional pooling layer is used to reduce the size to a scalar value. The last three layers are fully connected layers. After each convolution a dropout and a batch normalization is performed. Every layer except for the fully connected layer is followed by a leaky ReLu activation. The activation used after the last layer is the sigmoid function to convert the output to a value between 0 and 1 for the binary classification.

The convolution layers have a kernel size of (3x3x3) with a stride of (1x2x2) and a padding of one. By using a different stride for the first dimension, only the second and

Table 4.1: Overview of the layers in the CNN

Layer	input channels	output channels	input shape	output shape
Convolution	1	16	64x256x256	64x128x128
MaxPool	16	16	64x128x128	32x64x64
Convolution	16	32	32x64x64	32x32x32
MaxPool	32	32	32x32x32	16x16x16
Convolution	32	64	16x16x16	16x8x8
MaxPool	64	64	16x8x8	8x4x4
Convolution	64	128	8x4x4	8x2x2
MaxPool	128	128	8x2x2	4x1x1
MaxPool	128	128	4x1x1	2x1x1
MaxPool	128	128	2x1x1	1x1x1
Fully Connected	128	64	1x1x1	1x1x1
Fully Connected	64	32	1x1x1	1x1x1
Fully Connected	32	1	1x1x1	1x1x1

third dimension are downsampled in the convolution layers. This is needed, because the first dimension has a smaller size from the beginning and thus needs to be downsampled less often. The first convolution layer has one input channel and 64 output channels. Every following convolution doubles the amount of channels, resulting in 512 output channels in the last convolution layer.

The pooling layers use symmetrical kernels of size (2x2x2) with a stride of (2x2x2). This results in the size being divided by two in each dimension. An exception is the last pooling layer, where the size of the last two dimensions is already reduced to one and only the first dimension needs to be downsampled further. In this layer, a kernel of size (2x1x1) and stride (2x1x1) is used, which has no effect on the last two dimensions. Pooling layers have no effect on the amount of channels.

The last layers are three fully connected layers. They gradually reduce the number of channels to one. Thus, the shape of the CNNs output is $(n, 1, 1, 1, 1)$, where n is the number of samples that were passed through the network simultaneously.

The training of the network is performed in multiple epochs. In each epoch, all samples are passed through the network in several batches. The first training step for each batch, is to calculate the models predictions for the samples in the batch. Those predictions are then compared to the labels given to the samples. The loss function calculates the distance between the predicted and the actual labels. For binary classification the binary cross entropy function is suitable. PyTorch offers a loss function that connects the BCE loss with the sigmoid activation, meaning that the activation of the last layer in the network can be dropped. By combining the two steps in one function,

the operation becomes more numerically stable. BCE loss also allows for weights to be assigned to the classes. Before this loss can be passed backwards through the network to adjust the weights, the gradients of all optimized tensors need to be reset. Otherwise, the gradients would accumulate and add up with the current batch, leading to incorrect weight updates. After the loss is backwards passing of the loss is finished the models parameters are updated. The updating is done by the optimizer. For this work, Adam was chosen as the optimizer. After this is repeated for each batch the epoch is finished.
The trainings success is influenced by the chosen hyper parameters. These include the learning rate of the optimizer, the batch size, the number of epochs, the dropout rate and the weight assigned to the positive class. The optimal parameters for training are determined by a parameter analysis. The best results were achieved by the following parameter combination:

- Learning Rate: 0.00001
- Number of Epochs: 30
- Batch Size: 8
- Dropout Rate: 0.3
- Weight of Positive Class: 1.3

After the training is done, the model is evaluated. For this, the model calculates predictions for samples it was not trained on. These predictions are then compared to the actual labels of the samples and several performance metrics are determined. To get a detailed estimation of the performance, the calculated metrics are: accuracy, precision, recall, F1 score and the confusion matrix. In the case of predictive maintenance, a false negative would mean, that a malfunction is not predicted and can therefore not be fixed before the component breaks. This can potentially be expensive and dangerous, making it the more severe error. Thus, the objective is to maximize the recall.

5 Results

This Chapter delves into the results of the two test campaigns and the subsequent vibration analysis. Firstly, section 5.1 covers the measurement results of two test campaigns and the incidents that occurred during them. In the second section 5.2, the results of the vibration analysis and the machine learning model are explained.

5.1 Measurement Results of the Test Campaigns

Over the course of this work, the vibrations during the life cycles of two different types of REPAs were analyzed. The data was obtained during two test campaigns. The first test campaign featured RFHA as pipe connectors and was performed in 2024 before this thesis was started. Section 5.1.1 covers the vibration measurements that were performed during this campaign.
The second test campaign, which features BJAs was performed in 2025 during the course of this work. The measurements and incidents of this experiment are topic of section 5.1.2.

5.1.1 Rotary Flex Hose Assembly Test Campaign

During the RFHA test campaign, 13,723 cycles were completed at the cycling temperature of 389 °C. After 4,381 cycles, a problem with the traverse movement of the test rig occurred. While this is not due to a malfunction of the REPAs, it needs to be noted as it can also influence the vibrations that are measured by the sensors. After the problem was fixed, cycling continued until a small oil leakage was detected in the west swivel after 11,700 cycles. After 13,681 cycles, another leakage occurred in the west swivel which was accompanied by smoke. Cycling was continued for another 52 cycles and then stopped. Between the 500th and the 2204th cycle, no vibration data was collected.
Over the course of the test campaign, the vibration patterns changed. Figure 5.1 shows the spectrograms of the vibration data recorded during the 50th cycle in the beginning

of the test campaign in the east swivel. Figure 5.2 shows the spectrogram of the same cycle recorded in the west swivel. The x-axis of the plots contains the time elapsed

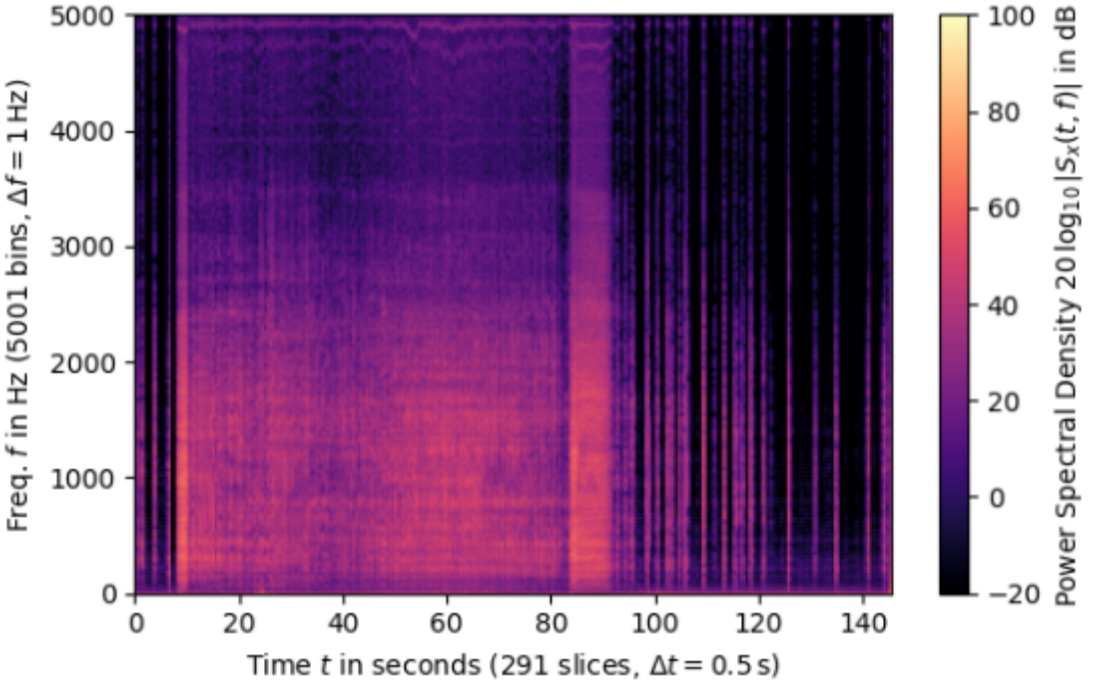

Figure 5.1: Spectrogram of the vibrations recorded on the east swivel during the 50th cycle

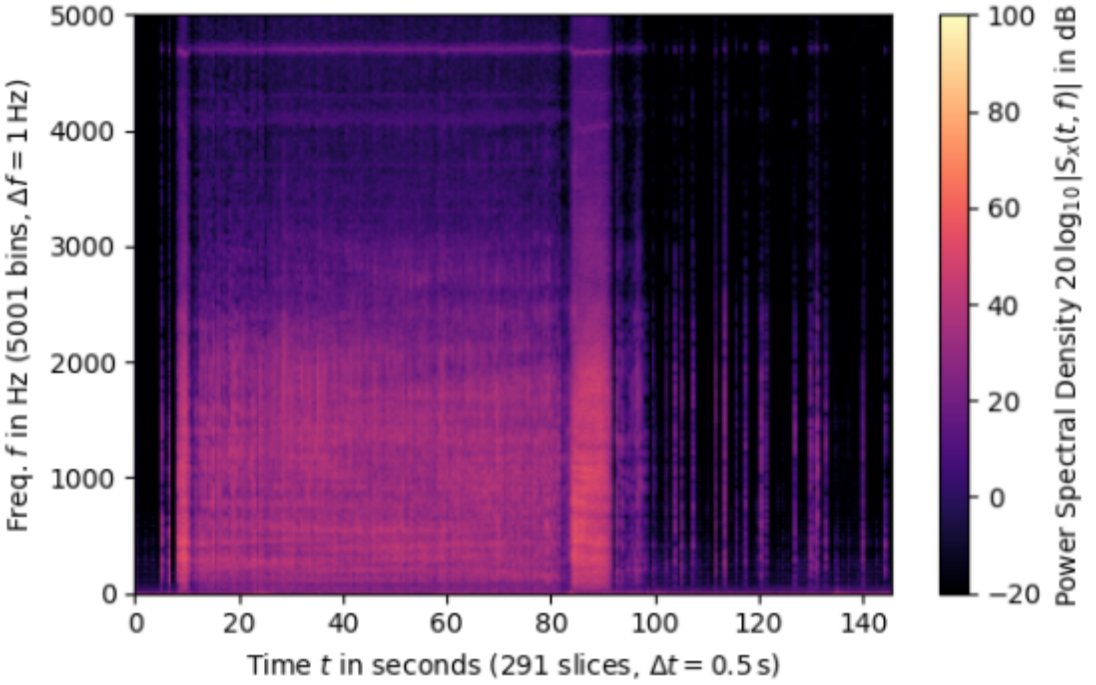

Figure 5.2: Spectrogram of the vibrations recorded in the west swivel during the 50th cycle

since the current cycle started in seconds, while the y-axis shows the frequency in Hz. Since the sampling frequency of the vibration measurement was 10,000 Hz, the Fourier transform can detect frequencies up to 5,000 Hz. The color bar shows the power spectral density in decibel. Since decibel are defined on a logarithmic scale, the color bar shows values below zero even though the power spectral density is always positive.

The spectrograms of the data collected by the two sensors include similar patterns. Both have a window of higher power spectral densities between 10 and 90 seconds. In both cases this window starts and ends with a peak of the power spectral density. Furthermore, the spectrograms show higher power spectral densities for lower frequencies up to about 2,500 Hz and lower power spectral densities for frequencies above this threshold.

Figure 5.3 and 5.4 show the spectrograms of the vibration data recorded during the 6000th cycle in the east and the west swivel. This cycle happened after the problem with the traverse was fixed. After this cycle the REPAs show no sign of malfunction for another 5,700 cycles and thus belong to the class of cycles that do not point to a failure that is about to happen.

The patterns in these spectrograms differ from the ones shown in Figures 5.1 and5.2 that were recorded 5,950 cycles earlier. They include two windows with elevated power spectral densities. In the spectrogram of the data recorded on the east swivel, these windows occur between the 20th and the 58th and the 83rd and the 138th second.

In the data recorded by the sensor on the west swivel, similar patterns can be observed. However, they occur at slightly different times. Here, the windows of higher power spectral densities are between the 22nd and the 47th and the 97th and the 132nd second. Additionally, at very low frequencies elevated power spectral densities appear before the second window from the 79th second onward. The spectrogram of the data recorded on the east swivel features higher power spectral density peaks than the one of the data recorded on the west swivel.

Overall, the power spectral density does not vary as strongly over the frequencies as compared to the spectrograms of the 50th cycle. Additionally, there are big regions with very low spectral power densities between the windows mentioned above that do not appear in the spectrograms of the 50th cycle. The link between higher frequencies and higher power spectral densities that was present in the spectrograms in Figures 5.1 and 5.2 can still be observed in these spectrograms.

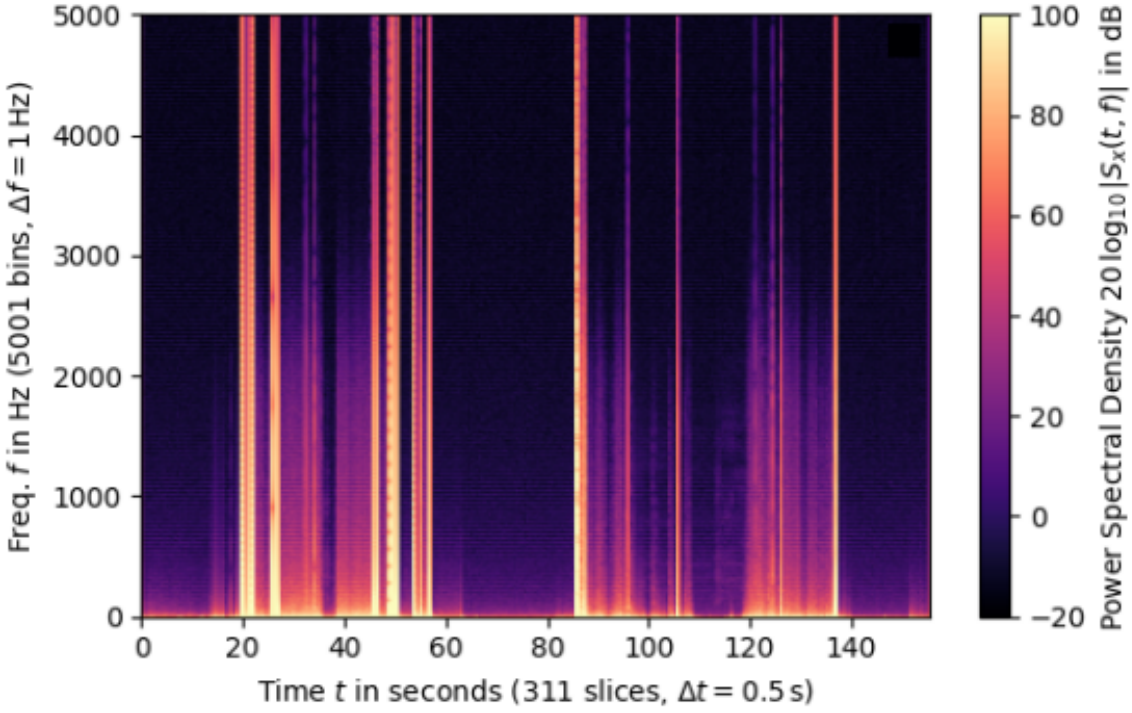

Figure 5.3: Spectrogram of the vibrations recorded on the east swivel during the 6000th cycle

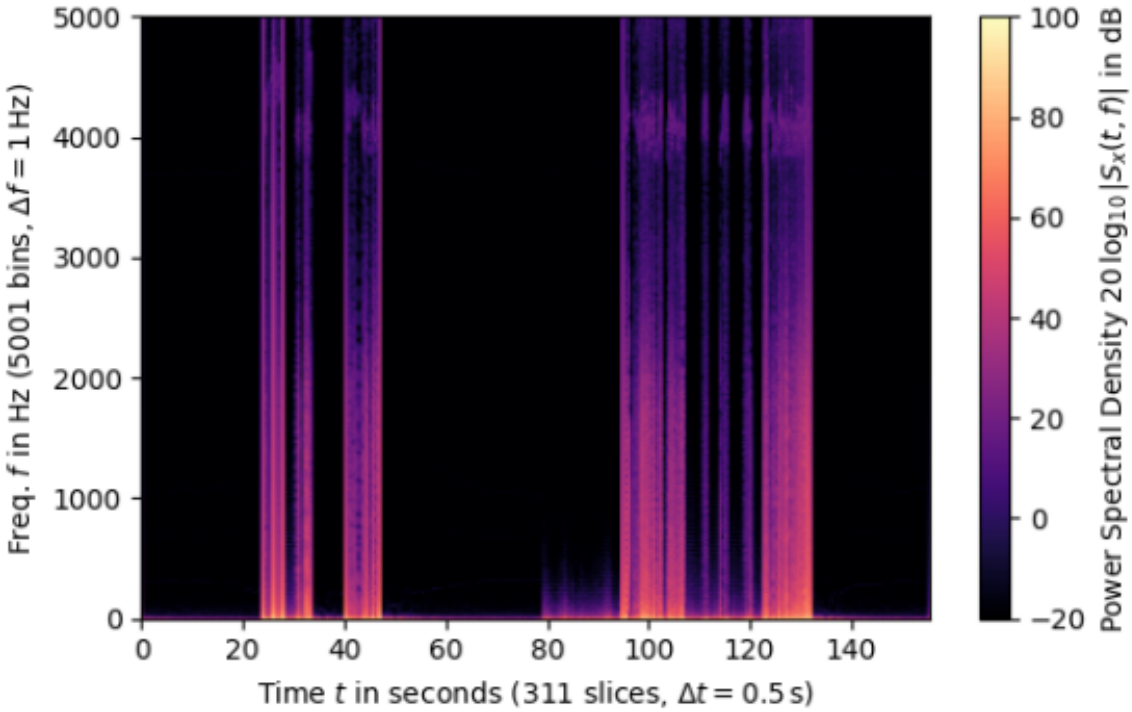

Figure 5.4: Spectrogram of the vibrations recorded in the west swivel during the 6000th cycle

In conclusion, the spectrograms of the 6,000th cycle look quite different to the ones of the 50th cycle. However, they all belong to the negative class. While there are some differences between the spectrograms of the data recorded by the two sensors in the 6,000th cycle, similar patterns can be observed in both.
In Figures 5.5 and 5.6, the spectrograms of the data recorded during the 13,500 rotation cycle are pictured. Figure 5.5 shows the data of the east swivel, while 5.6 shows the data recorded in the west swivel. About 200 cycles after this, the leakage occurred in the west swivel and the test campaign was stopped. Thus, these cycles are part of the positive class.
In the spectrogram in Figure 5.5, similar patterns as in the spectrograms in Figure 5.3 can be observed. The first window of higher spectral densities featured in Figure 5.3 is now split into two peaks at 19 and at 30 seconds that last for about one second. Between these peaks, the power spectral density is very low. At the end of the cycle, a window of elevated power spectral densities is present at a similar spot as in the data of the 6000th cycle. In the other regions the power spectral density is constantly very low. In this spectrogram it starts at the 87th second and ends at the 135th second. Overall, this spectrogram also shows higher spectral power densities in lower frequencies.
The spectrogram of the data recorded in the west swivel (figure 5.6) does not feature this pattern of low power spectral densities with peaks and windows of higher values in between. Here, the power spectral density is constant over all frequencies and times.

This big difference between the spectrograms of the data recorded by the two sensors of the same cycle can only be observed for the cycles in the positive class, which is the class of cycles leading up to a malfunction. The spectrograms of the data recorded in different cycles are not always similar even though they belong to the same class. An example of this are the spectrograms shown in the Figures 5.1 and 5.3, which are both recorded in the east swivel. While they both belong to the negative class they do not include similar patterns.
To explore this tendency further, the SSIM of the two spectrograms was calculated for every cycle, as explained in section 4.1. The result of this is pictured in Figure 5.7, where the SSIM comparing the two spectrograms of each cycle is plotted. The x-axis shows the cycle number, while the y-axis shows the SSIM on a scale from 0 to 1. A score of 0 would mean, that the two spectrograms are completely different, while a score of 1 is given to identical spectrograms. The Figure also includes the cycle num-

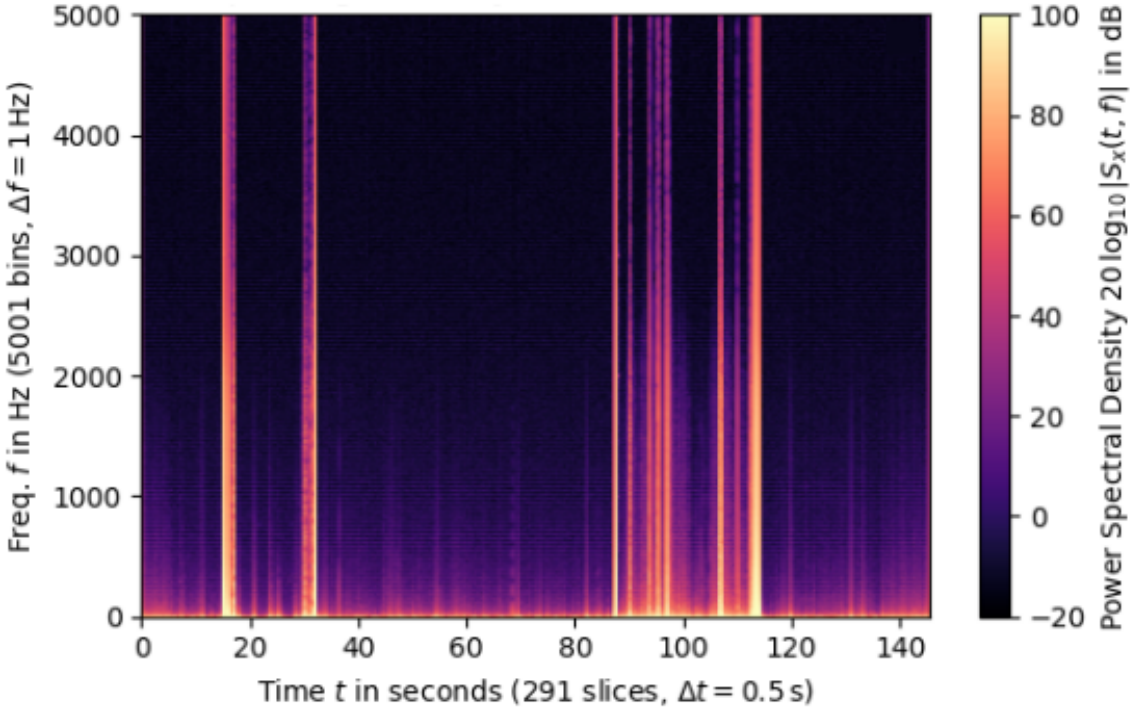

Figure 5.5: Spectrogram of the vibrations recorded on the east swivel during the 13,500th cycle

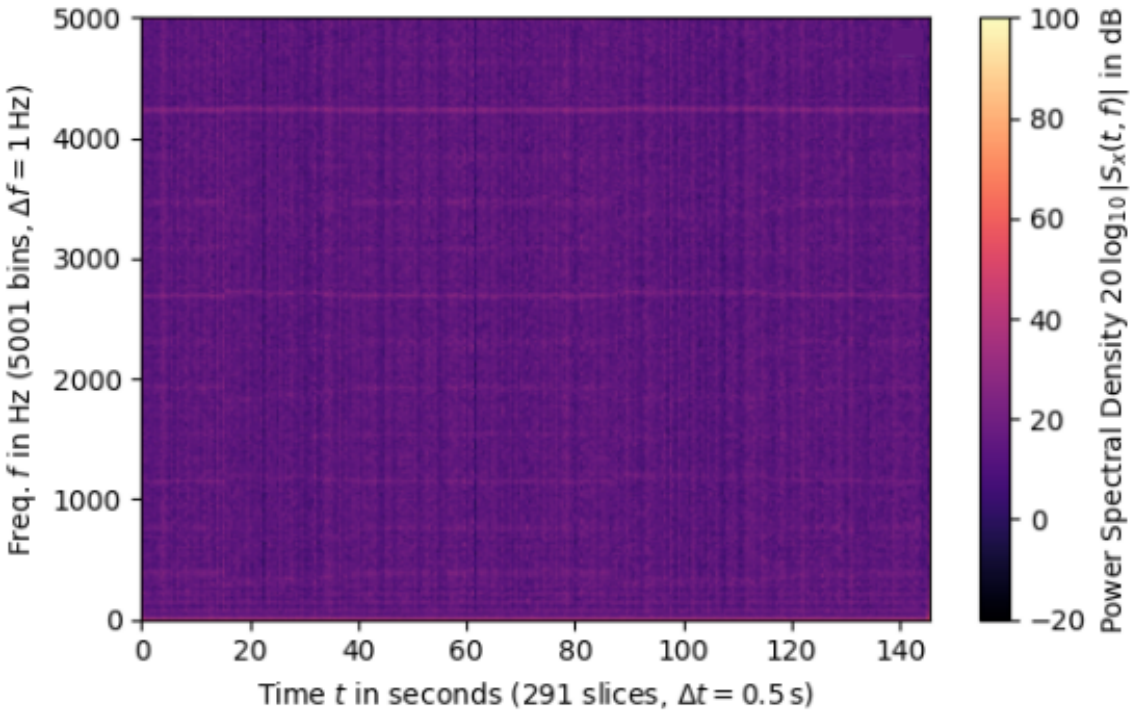

Figure 5.6: Spectrogram of the vibrations recorded in the west swivel during the 13,500th cycle

ber of the three incidents that happened during the test campaign. This is shown in red. Between the 500th and the 2200th cycle there is a gap, as no data was recorded. In the ares where no incident is close, very high similarity scores of nearly one were calculated. In the first few cycles the score is slightly lower which can be explained by the higher variation of spectral power densities in the spectrograms of these cycles. This can be seen in Figures 5.1 and 5.2. In later cycles, for example in the cycles pictured in Figures 5.3 and 5.4 there are larger areas with no variation which leads to higher similarity scores. Before a malfunction occurs, a strong drop of the similarity score can be observed.

Before the problem with the traverse happened, the similarity score was low for 2190 cycles. During this period, no cycle with a high similarity score was measured.

The second and third incident, which were both problems with the west swivel, were led by a drop in the similarity score that lasted for about 1,000 cycles. In a real PTC which performs one rotation cycle per day, this translates to a little over 2.5 years. However, in the 1000 cycles leading up to the incidents, not every cycle showed a low similarity score. Especially before the final malfunction happened the similarity score went back to almost 1 for a few cycles before dropping again.

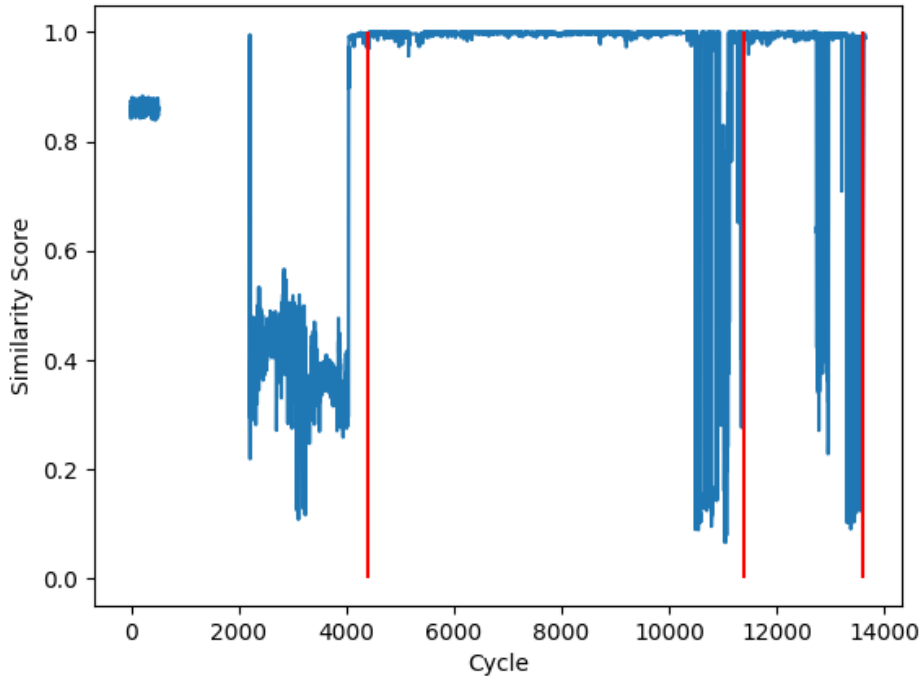

Figure 5.7: SSIM score comparing the spectrograms of the vibration data collected by sensors on each RFHA for every cycle, Red: Incidents happening during the test campaign

5.1.2 Ball Joint Assembly Tets Campaign

The second test campaign, which featured BJAs, was interrupted by a nationwide power outage after 2700 cycles were completed. Unfortunately, this caused the data that was collected up to this point to be lost. After the power supply was stable enough to start heating up again, outgassing was observed in five out of the six ball joints. To stop the smoking of the ball joints, the seals were refilled with graphite. However, more graphite than expected was needed to stop the outgassing. Additionally, the screws of several of the ball joints were damaged. The delays in the maintenance caused the test campaign to not be finished during this thesis.
After the power outage several test cycles were done to asses the need for maintenance and to investigate the change in the angles of the pipes between the ball joints. During these cycles, vibration data was collected. However, these cycles were done with a reduced rotation speed and with varying temperatures and pressures. Since no data of the well functioning ball joints is available, it is not possible to investigate which changes in the vibrations are due to the malfunctions and which are due to the changes in temperature, pressure and rotation speed. The varying rotation speed causes the differences in the length of the spectrograms shown in this section, as the cycles are longer if the REPA is rotated more slowly. The cycle length was varied from 125s to 295s.
The spectrogram of the vibrations collected on the lower ball joint of the west REPA is pictured in Figure 5.8. It shows similarities to the spectrograms collected in the RFHA test campaign, e.g. the ones in figures 5.3 and 5.4. This suggests, that some of the collected vibrations originated in the hydraulic system.
Other cycles of the west BJA, as for example the one shown in Figure 5.9, lack these patterns. This spectrogram includes no areas of higher spectral densities and instead shows constant energies over all frequencies and times. This behavior was previously encountered in the west swivel in the RFHA test campaign before the outgassing was observed. However, to determine if these patterns occurred again due to the malfunction of the ball joint, more data of the well functioning ball joint is needed.
Examples of the spectrograms collected on the eastern BJA are shown in Figures 5.10 and 5.11. The first pictured spectrogram features windows of higher power spectral densities, most notably between the 150th and the 200th second of the cycle. The second spectrogram is similar to the second spectrograms that was shown of the western

BJA with constant power spectral densities over all times and frequencies.

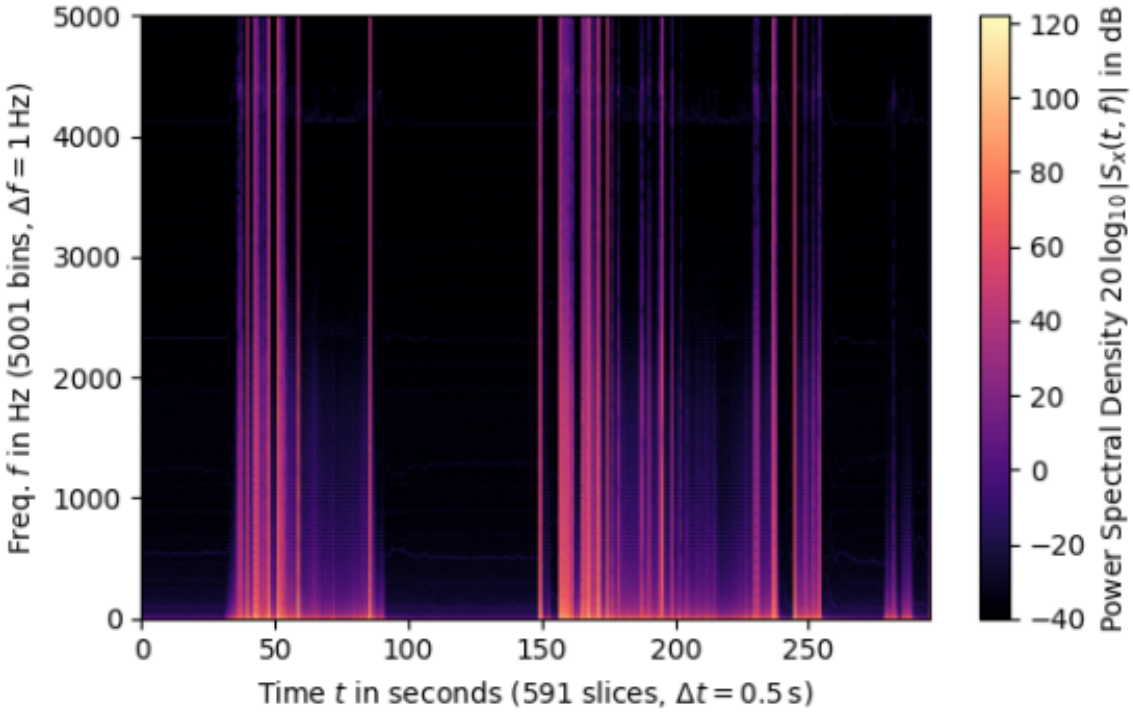

Figure 5.8: Spectrogram of the vibrations recorded on the lower western ball joint during a slow test cycle after the first 2,700 cycles were completed

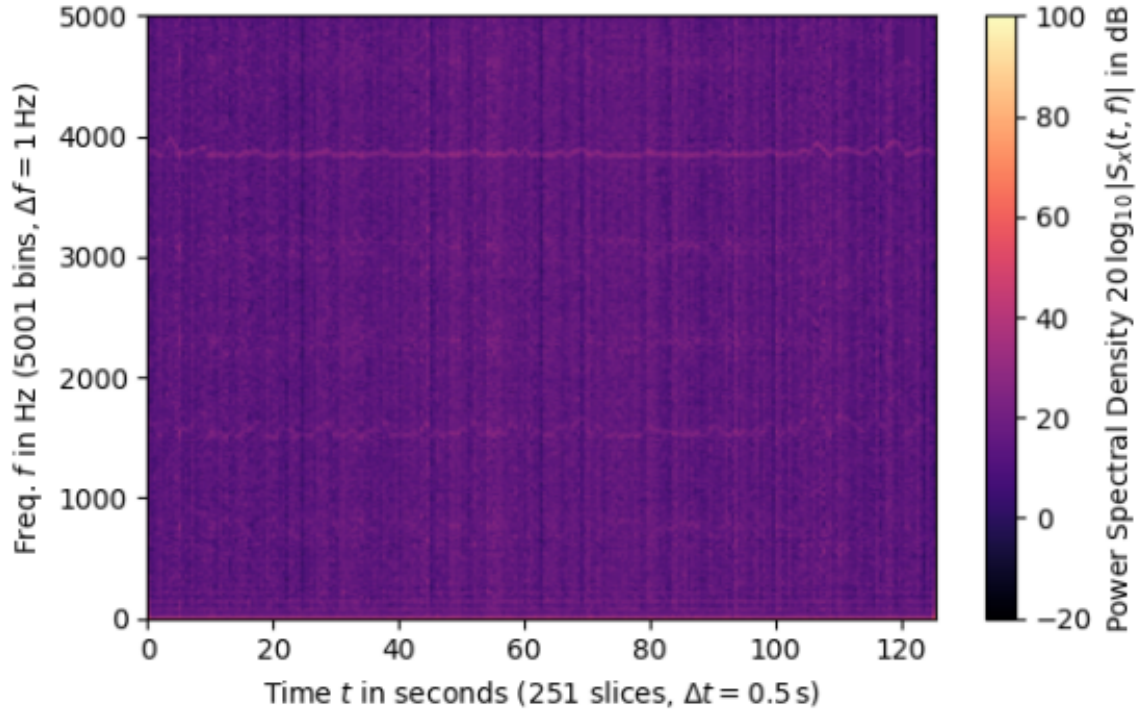

Figure 5.9: Spectrogram of the vibrations recorded on the lower western ball joint during a fast test cycle after the first 2,700 cycles were completed

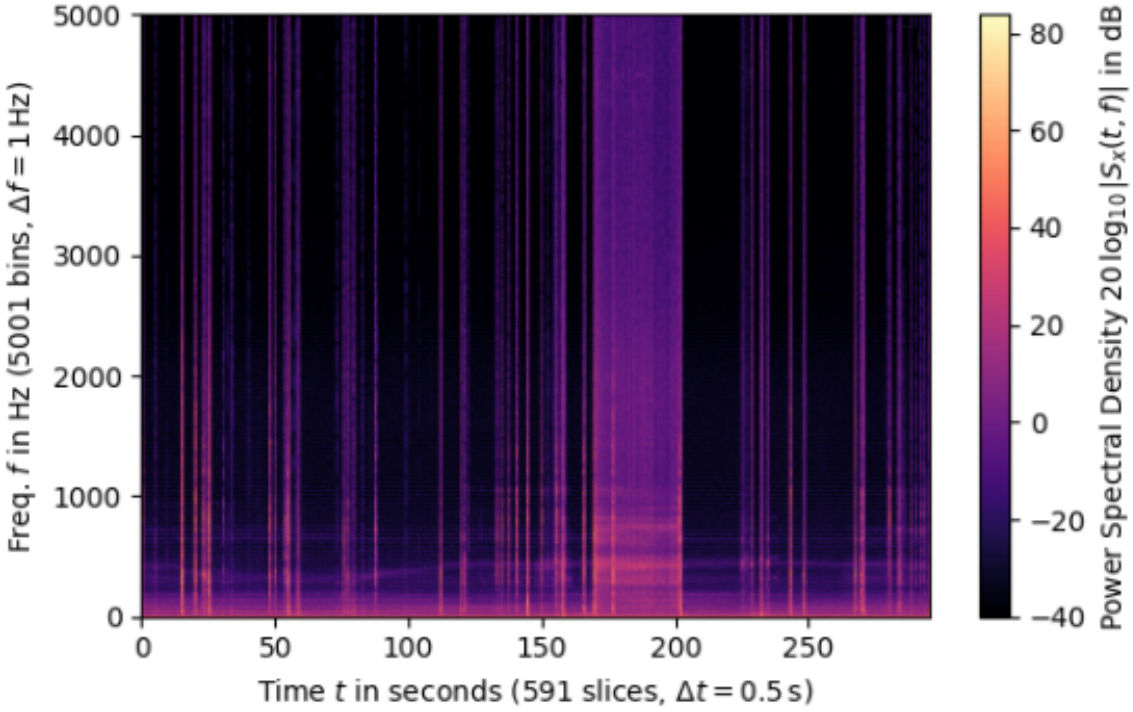

Figure 5.10: Spectrogram of the vibrations recorded on the lower eastern ball joint during a slow test cycle after the first 2,700 cycles were completed

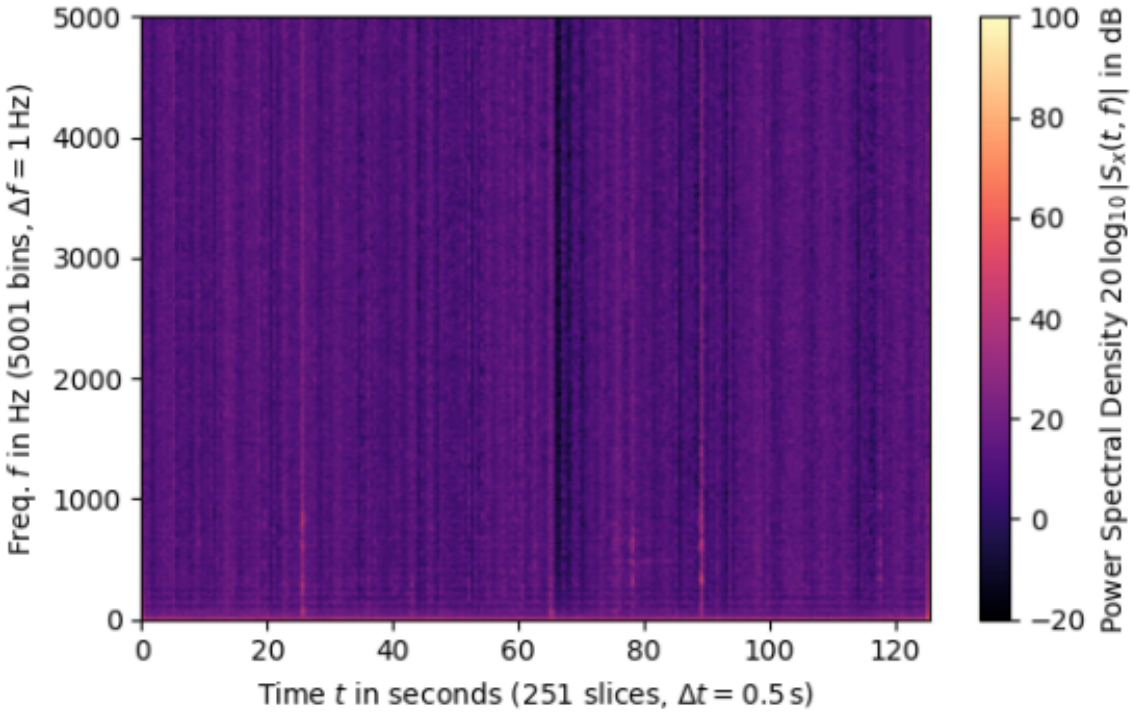

Figure 5.11: Spectrogram of the vibrations recorded on the lower eastern ball joint during a fast test cycle after the first 2,700 cycles were completed

5.2 Results of the Machine Learning Methods

As described in section 4.2, a CNN was trained to recognize comparison spectrograms of cycles that precedented a malfunction of the REPAs. The model was trained on the data collected in the RFHA test campaign, since the data collected in the BJA test campaign was not large enough to perform training.
Figure 5.12 shows the failure probability over the entire life cycle of the RFHAs calculated by the trained model. The Figure includes the time at which each of the three problems of the test campaign occurred. It should be noted, that in this case the model was trained on the data that was recorded before the second malfunction occurred. Thus, the performance of the model can only be correctly estimated based on the predictions made for the data after the second problem. This was used as the testing dataset for the model.

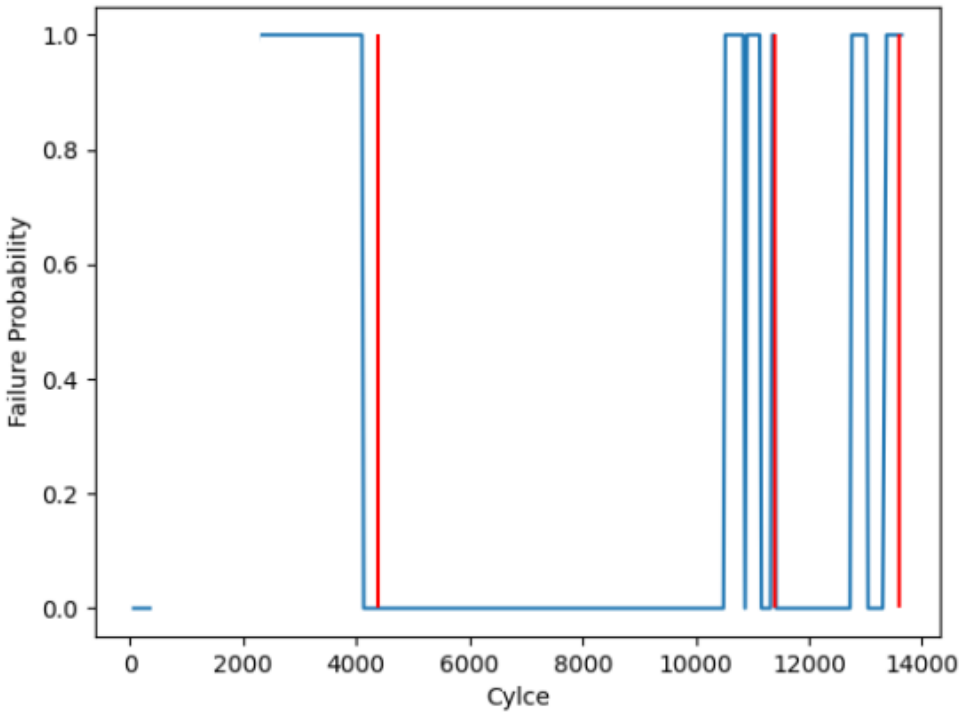

Figure 5.12: Failure Probability over the life cycle of the RFHA, Red: Incidents happening during the test campaign

The figure shows, that before a failure happens the predicted failure probability also rises. Before the two leakages (failure two and three), the time frame in which an increased failure probability is predicted is similar. In both cases the failure probability is one for the first time 1000 cycles before the malfunction occurs.

In the 1000 cycles before the second and third problem happened, the failure probability does not stay at one but drops to zero again several times. This can be explained by the variance that can be observed in the similarity score in these areas (compare Figure 5.7). The method of calculating the probability for 64 cycles at once balances out some of the variance, making the probability more stable than the similarity score. In some areas however this is not enough to guarantee that the probability stays at one for the entire period before the failure.
During the periods where no failure is about to happen the probability stays at zero. This is due to the low variance in the similarity score in these areas.
To quantify the models performance, several performance metrics were calculated:

- Accuracy: 0.93
- Precision: 1
- Recall: 0.82
- F1 Score: 0.9
- True Negative Rate: 1
- True Positive Rate: 0.82
- False Negative Rate: 0.18
- False Positive Rate: 0

The precision of one can be explained by the low variance in the areas where no malfunction is to be expected. The false negative rate corresponds to the drops in the failure probability that occurs right before the second and third problem in Figure 5.12.

6 Discussion

As shown in chapter 5.1.1, a correlation between the SSIM score calculated of the two spectrograms of each cycle and the closeness to a RFHA malfunction was found. By using this method, a CNN was able to predict failures at least 1000 cycles before they occurred. With one cycle lasting one day in a real power plant, this translates to a failure prediction more than 2.5 years in advance, making predictive maintenance possible. Predictive maintenance lowers the operation cost of the plants by minimizing downtime and making the system more reliable. Additionally the system safety is improved by the predictive maintenance, since leakage of the HTF can lead to fires in the power plants.

However, there are several differences between the test rig and a real PTC power plant that have to be considered in order to transfer the findings of this thesis to the real application. Firstly, the rotation movement in the test rig is always done fast and continuous. In a PTC power plant, there are three different rotation movements: from the stow position to the starting angle, from the starting angle to the end angle and from the end angle back to the starting position. While the first and the last of these movements are done continuously, the movement from the starting angle to the end angle is slow and incremental. The lower speed and the starting and stopping might have an impact on the vibrations that can not be analyzed from the data collected in this thesis. Furthermore, the hydraulic system that is responsible for the rotation movement is in a different position in the test rig as compared to a real PTC. The similarities between the vibration patterns of the BJA and the RFHA test campaigns suggest, that the vibrations of the hydraulic system were also measured by the sensors. These patterns might look different in a real PTC power plant, since the kinematic systems are set up differently.

Another factor that has to be considered is the difference in the length of the life cycles. While in the test rig 10,000 cycles can be completed in less than a month, in a real PTC plant, the REPAs are expected to last for 25-30 years. In this time span environmental factors can change, also influencing the vibrations. These environmental factor include wind, temperature or sun intensity. Especially wind is important in this consideration, since it can change rapidly, making even the comparison of cycles that happened subsequently difficult. The proposed method of comparing the data collected in both REPAs rather than comparing different cycles aims to prevent this effect. However, the

extend to which the longer lifespan effects the development of the vibrations over one life cycle has to be investigated in future research.
Additionally, in the test rig, the expansion of the receiver tube was simulated by adding translational movement of the kinematic unit. This translational movement was kept constant over the course of the test campaign. In a real plant however, the heat expansion of the receiver tube would not be the same in every cycle, since it depends on the achieved HTF temperature. Since the sun exposure is not constant every day, the HTF does not reach the maximal temperature in every cycle. The differences in translational movement might also impact the recorded vibrations.
The temperature of the HTF and therefore also the expansion of the receiver tube is additionally impacted by the chosen HTF. The maximum temperature is different for every HTF, which has to be considered when interpreting the vibrations. Since some HTFs are present in two different phases in the tubing system, the phase change might impact the vibrations of the REPAs. The choice of HTF also impacts the required pressure in the tubing system. In this thesis, only *Therminol ®VP-1* oil was tested.
Since of each type of REPA only one life cycle was analyzed, it is not possible to detect which part of the vibration patterns are generally applicable to all life cycles and which are specific to this singular experiment. To extract the parts of the data that are generally applicable, more data needs to be collected.
Due to the power outage and the following data loss, not enough data was collected of the BJA to train a classification model. To investigate the applicability of the proposed classification model for BJAs, more data needs to be acquired.
The thesis was successful in proving, that there are changes in the vibrations that indicate REPA failure. A predictive maintenance strategy was proposed that predicts REPA failure based on the similarity of the vibrations occurring in the REPAs on both sides of a PTC. This prevents the influence of environmental effects such as wind and makes the strategy more generally applicable.

7 Conclusion

The objective of this work was to analyze the vibrations occurring in the REPAs of PTC power plants to find patterns that point to a future failure. This could potentially be used to implement a predictive maintenance strategy in the power plants. Since the REPAs carry the hot HTF, unexpected failures can potentially lead to fires. Thus, a predictive maintenance strategy can make the operation of the PTCs safer. Additionally, predictive maintenance enables the efficient scheduling of maintenance and therefore of the downtimes, saving the cost of long unexpected breaks in operation.

Two different kinds of REPAs were investigated. In the first test campaign, which was performed before this thesis was started, rotary flex hose assemblies (RFHA) were tested. The second test campaign featured ball joint assemblies (BJA). This test campaign could not be completed before the end of this thesis.

Both test campaigns were performed at the REPA test rig at the Plataforma Solar de Almería in Spain. The test rig can be used to analyze the life cycle of REPAs in a shorter time span by simulating up to 570 rotation cycles per day. Thus the estimated lifespan of 10,000 cycles can be achieved in less than a month. Two REPAs are tested simultaneously, which is compareable to a real PTC where there is a REPA on each side of the collector. The test rig performs rotational and translational movement simultaneously. The translational movement imitates the heat expansion of the receiver tube in a real PTC.

The vibrations of the REPAs were recorded using piezoelectric accelerometers. They were analyzed using short time Fourier transform to create spectrograms of the frequencies present in the signals. For each rotational cycle, two spectrograms were created, one for each side of the test rig. To lower the impact of local variations and to make the predictions more general, a comparison picture of the two spectrograms recorded of each cycle was created using the structural similarity index metric [73]. The comparison pictures were used to train a CNN to distinguish between the vibrations occurring in REPAs that functioned normally and the ones occurring in REPAs that were about to fail.

During the first test campaign, 13,723 cycles were completed at the cycling temperature of 389 °C, before a HTF leakage in the west swivel caused the campaign to be stopped. Three problems occurred over the course of the life cycle. Each of the prob-

lems was led by a drop in the similarity score. An explanation for this is, that regularly functioning REPAs will show similar vibrations when performing the same movement simultaneously. However, if one REPA malfunctions, the vibrations change and differ from the ones measured in the intact REPA. The CNN trained to detect the comparison pictures of the spectrograms occurring shortly before a failure was able to predict the failures 1,000 cycles in advance. With one rotation cycle happening every day, this means, that failure could be predicted more than 2.5 years in advance in a real power plant.

The second test campaign was interrupted by a nation wide power outage. When it was resumed, outgassing could be observed in five out of the six ball joints. Before the power outage, 2,700 cycles were already completed, which is equivalent to seven years of operation in a real power plant. Since seven years is the typical time after which maintenance in the form of regraphiting has to be done in a BJA, this was expected. However, as a result of the power outage, the vibration data collected up to that point was lost. Since the time required for the maintenance of the ball joints was longer than expected, the campaign could not be completed. The data that was analyzed of this test campaign was collected during testing cycles that were done after the power outage before maintenance was performed. In the spectrograms calculated from these data similar patterns as in the RFHA test campaign could be observed, suggesting that the hydraulic system influences the vibrations. The data collected in this campaign was not large enough to be used in the training of a CNN.

Since there are many differences between the test rig and an actual PTC power plant, further research has to be done on the vibration behavior of REPAs before the predictive maintenance strategy can be employed. These differences include different movement speeds, life cycle length, environmental factors such as wind, and differences in the setup of the kinematic system. Furthermore, more data of the BJAs has to be collected to verify, that the prediction method suggested for the RFHA can be applied to the BJAs as well.

Due to these limitations, this work has to be understood as a proof of concept of using vibration data for the predictive maintenance of REPAs in PTCs. It shows, that installing vibration sensors in PTC power plants is beneficial for a safe and cost efficient operation.

In future works, more data could be collected, especially on the BJAs. When more data is available, the classification model could be expanded to analyze where in the

spectrogram comparison image the differences occur. This could be used to determine what kind of failure will occur or how many functional cycles are left until the REPA will malfunction.

Bibliography

1. KLENIN, M. *Analysis of mechanically induced noise on flexible pipe connectors for parabolic trough solar collectors*. 2022. Masters Thesis. Tongji University.
2. UNITED NATIONS TREATY COLLECTION. *Paris Agreement*. 2015-12.
3. INTERNATIONAL ENERGY AGENCY. *Electricity Report 2025*. 2025.
4. INTERNATIONAL ENERGY AGENCY. *Solar PV* [online]. [visited on 08/05/2025]. Available from: https://www.iea.org/energy-system/renewables/solar-pv.
5. FERREIRA DA SILVA, M. et al. Audible Noise Evaluation in Wind Turbines Through Artificial Intelligence Techniques. *Sensors*. 2025, vol. 25, no. 5. Available from DOI: 10.3390/s25051492.
6. RANJAN, R.; GHOSH, S. K.; KUMAR, M. Fault diagnosis of journal bearing in a hydropower plant using wear debris, vibration and temperature analysis: A case study. *Proceedings of the Institution of Mechanical Engineers, Part E*. 2020, vol. 234, no. 3, pp. 235–242. Available from DOI: 10.1177/0954408920910290.
7. ALAMI, A. H. et al. Concentrating solar power (CSP) technologies: Status and analysis. *International Journal of Thermofluids*. 2023-05, vol. 18, p. 100340. Available from DOI: 10.1016/j.ijft.2023.100340.
8. WEINSTEIN, L. et al. Concentrating Solar Power. *Chemical reviews*. 2015-10, vol. 115. Available from DOI: 10.1021/acs.chemrev.5b00397.
9. SOLARGIS. *Solar resource maps of World* [online]. [visited on 08/04/2025]. Available from: https://solargis.com/resources/free-maps-and-gis-data.
10. AHMAD, A. et al. Parabolic trough solar collectors: A sustainable and efficient energy source. *Materials Science for Energy Technologies*. 2024, vol. 7, pp. 99–106. ISSN 2589-2991. Available from DOI: 10.1016/j.mset.2023.08.002.
11. MOUKHTAR, I. et al. *Solar Energy: Technologies, Design, Modeling, and Economics*. Springer Cham, 2020-11. ISBN 978-3-030-61307-5. Available from DOI: 10.1007/978-3-030-61307-5.
12. PRICE, H. et al. Advances in Parabolic Trough Solar Power Technology. *Journal of Solar Energy Engineering*. 2002-04, vol. 124, no. 2, pp. 109–125. Available from DOI: 10.1115/1.1467922.

13. BURKHARDT, J. J.; HEATH, G. A.; TURCHI, C. S. 'Life Cycle Assessment of a Parabolic Trough Concentrating Solar Power Plant and the Impacts of Key Design Alternatives. *Environmental Science & Technology*. 2011, pp. 2457–2464. Available from DOI: 10.1021/es1033266.

14. PLUMPE, A. *Design of a Test Rig and its Testing Methods for Rotation and Expansion Performing Assemblies in Parabolic Trough Collector Power Plants*. 2016. Masters Thesis. RWTH Aachen Institute of Solar Research, German Aerospace Center (DLR).

15. SCHNEIDER, F. *Commissioning and optimization of a test bench for life cycle Analysis of Rotation and Expansion performing assemblies (REPAs) in parabolic trough collector power plants*. 2019. Masters Thesis. RWTH Aachen Institute of Solar Research, German Aerospace Center (DLR).

16. SUSTO, G. A. et al. Machine Learning for Predictive Maintenance: A Multiple Classifier Approach. *Industrial Informatics, IEEE Transactions on*. 2015-06, vol. 11, pp. 812–820. Available from DOI: 10.1109/TII.2014.2349359.

17. MOBLEY, R. K. *An Introduction to Predictive Maintenance*. 2nd. New York, NY: Elsevier Inc., 2002. ISBN 978-0-7506-7531-4. Available from DOI: 10.1016/B978-0-7506-7531-4.X5000-3.

18. TAYLOR, E. F.; WHEELER, J. A. *Spacetime Physics*. W. H. Freeman and Company, 1966. ISBN 0-7167-0336-X.

19. EINSTEIN, A. *On the Relativity Principle and the Conclusions Drawn from It*. 1907. Available from DOI: 10.1093/ajae/aaq063.

20. DOSCHER, J. *Accelerometer Design and Applications*. 2008.

21. CURIE, J.; CURIE, P. Développement par compression de l'électricité polaire dans les cristaux hémièdres à faces inclinées. *Bulletin de Minéralogie*. 1880, vol. 3, no. 4, pp. 90–93. Available from DOI: 10.3406/bulmi.1880.1564.

22. MARTIN, R. M. Piezoelectricity. *Phys. Rev. B*. 1972-02, vol. 5, pp. 1607–1613. Available from DOI: 10.1103/PhysRevB.5.1607.

23. GAUTSCHI, G. *Piezoelectric Sensorics, Force Strain Pressure Acceleration and Acoustic Emission Sensors Materials and Amplifiers*. Springer Berlin, Heidelberg, 2002. ISBN 978-3-540-42259-4. Available from DOI: 10.1007/978-3-662-04732-3.

24. PIECOELECTRONICS, P. *Introduction to ICP Accelerometers*. Available also from: https://www.pcb.com/resources/technical-information/introduction-to-accelerometers.

25. FOURIER, J. B. J. *Théorie Analytique de la Chaleur*. Firmin Didot, 1822.

26. PINSKY, M. *Introduction to Fourier Analysis and Wavelets*. Brooks/Cole, 2002. ISBN 978-0-534-37660-4.

27. MÜLLER, M. Fourier Analysis of Signals. In: *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Cham: Springer International Publishing, 2015, pp. 39–114. ISBN 978-3-319-21945-5. Available from DOI: 10.1007/978-3-319-21945-5_2.

28. OPPENHEIM, A. V.; SCHAFER, R. W.; BUCK, J. R. *Discrete-time signal processing*. 2nd. Upper Saddle River, N.J. : Prentice Hall, 1999. ISBN 0-13-754920-2.

29. ALLEN, J. Short term spectral analysis, synthesis, and modification by discrete Fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. 1977, vol. 25, no. 3, pp. 235–238. Available from DOI: 10.1109/TASSP.1977.1162950.

30. SAMUEL, A. L. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*. 2000, vol. 44, no. 1.2, pp. 206–226. Available from DOI: 10.1147/rd.441.0206.

31. MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. *Foundations of Machine Learning*. 2nd ed. Cambridge, MA: The MIT Press, 2018. ISBN 978-0262039406.

32. HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. Unsupervised Learning. In: *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY: Springer New York, 2009, pp. 485–585. ISBN 978-0-387-84858-7. Available from DOI: 10.1007/978-0-387-84858-7_14.

33. LIAO, W.; MOSEMAN, A. Developing a Reinforcement Learning based Chess Engine. *Proceedings of the West Virginia Academy of Science*. 2023-04, vol. 95. Available from DOI: 10.55632/pwvas.v95i2.990.

34. REN, Y. et al. Self-Learned Intelligence for Integrated Decision and Control of Automated Vehicles at Signalized Intersections. *IEEE Transactions on Intelligent Transportation Systems*. 2022, vol. 23, no. 12. Available from DOI: 10.1109/TITS.2022.3196167.

35. SALAZAR DUQUE, E. M. et al. Community energy storage operation via reinforcement learning with eligibility traces. *Electric Power Systems Research*. 2022, vol. 212. ISSN 0378-7796. Available from DOI: 10.1016/j.epsr.2022.108515.

36. RASCHKA, S.; MIRJALILI, V. *Python Machine Learning, 2nd Ed.* 2nd ed. Birmingham, UK: Packt Publishing, 2017. ISBN 978-1787125933.

37. CORTES, C.; VAPNIK, V. N. Support-vector networks. *Machine Learning*. 1995, vol. 20, pp. 273–297. Available from DOI: 10.1007/BF00994018.

38. HO, T. K. Random Decision Forests. In: *Proceedings of the 3rd International Conference on Document Analysis and Recognition*. Montreal, QC: IEEE, 1995, pp. 278–282. Available from DOI: 10.1109/ICDAR.1995.598994.

39. EPANECHNIKOV, V. A. Non-Parametric Estimation of a Multivariate Probability Density. *Theory of Probability & Its Applications*. 1969, vol. 14, no. 1, pp. 153–158. Available from DOI: 10.1137/1114019.

40. ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*. 1958, vol. 65, no. 6, pp. 386–408. Available from DOI: 10.1037/h0042519.

41. SAMMUT, C.; WEBB, G. I. *Encyclopedia of Machine Learning*. 1st. New York, NY: Springer, 2010. ISBN 978-0-387-30768-8. Available from DOI: /10.1007/978-0-387-30164-8.

42. TERVEN, J. et al. Loss Functions and Metrics in Deep Learning. A Review. *Computer Science Review*. 2023-07. Available from DOI: 10.48550/arXiv.2307.02694.

43. SCHMIDHUBER, J. Deep learning in neural networks: An overview. *Neural Networks*. 2015, vol. 61, pp. 85–117. ISSN 0893-6080. Available from DOI: 10.1016/j.neunet.2014.09.003.

44. MCCULLOCH, W.; PITTS, W. A. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*. 1943, vol. 5, pp. 115–133. Available from DOI: doi.org/10.1007/BF02478259.

45. AGGARAWAL, C. C.; REDDY, C. K. *Data Classification: Algorithms and Applications*. 1st ed. CRC Press, 2015. ISBN 978-1-4665-8674-1.

46. HAYKIN, S. *Neural networks and learning machines*. 3rd ed. Ontario, Canada: Pearson Education, 2009. ISBN 978-0-13-147139-9.

47. WERBOS, P. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Science. Thesis (Ph. D.). Appl. Math.* 1974-01. PhD thesis. Harvard University.

48. GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. MIT Press, 2016.

49. LEHMANN, L.; CASELLA, G. *Theory of Point Estimation*. 2nd. New York, NY: Springer, 1998. ISBN 978-0-387-98502-2. Available from DOI: 10.1007/b98854.

50. GOODMAN, R.; MILLER, J. W.; SMYTH, P. *Objective Functions For Neural Network Classifier Design*. IEEE, 1991-06. Published. Available from DOI: 10.1109/ISIT.1991.695143.

51. ROSASCO, L. et al. Are Loss Functions All the Same? *Neural Computation*. 2004-05, vol. 16, no. 5, pp. 1063–1076. Available from DOI: 10.1162/089976604773135104.

52. LIN, T. et al. Focal Loss for Dense Object Detection. *CoRR*. 2017. Available from arXiv: 1708.02002.

53. S., K.; R. A., L. On Information and Sufficiency. *The Annals of Mathematical Statistics*. 1951-03, vol. 22, no. 1, pp. 79–86. Available from DOI: 10.1214/aoms/1177729694.

54. HAYKIN, S. *Neural Networks: A Comprehensive Foundation*. 1st. USA: Prentice Hall PTR, 1994. ISBN 0023527617.

55. HAWKINS, D. M. The Problem of Overfitting. *Journal of Chemical Information and Computer Sciences*. 2004, vol. 44, no. 1, pp. 1–12. Available from DOI: 10.1021/ci0342472.

56. SRIVASTAVA, N. et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*. 2014, vol. 15, no. 56, pp. 1929–1958. Available also from: http://jmlr.org/papers/v15/srivastava14a.html.

57. YAO, Y.; ROSASCO, L.; CAPONNETTO, A. On Early Stopping in Gradient Descent Learning. *Constructive Approximation*. 2007, vol. 26, pp. 289–315. Available from DOI: 10.1007/s00365-006-0663-2.

58. MAAS, A.; HANNUN, A.; NG, A. Rectifier Nonlinearities Improve Neural Network Acoustic Models. In: *Proceedings of the International Conference on Machine Learning*. Atlanta, Georgia, 2013.

59. HE, K. et al. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*. 2015. Available from arXiv: 1502.01852 [cs.CV].

60. RUDER, S. *An overview of gradient descent optimization algorithms*. 2017. Available from arXiv: 1609.04747 [cs.LG].

61. BOTTOU, L.; BOUSQUET, O. Optimization for Machine Learning. In: MIT Press, 2012, chap. The Tradeoffs of Large Scale Learning, pp. 351–368. ISBN 978-0-262-01646-9.

62. KINGMA, D. P.; BA, J. *Adam: A Method for Stochastic Optimization*. 2017. Available from arXiv: 1412.6980 [cs.LG].

63. LECUN, Y. et al. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*. 1989-12, vol. 1, no. 4, pp. 541–551. Available from DOI: 10.1162/neco.1989.1.4.541.

64. LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*. 2015-05, vol. 521, pp. 436–444. Available from DOI: 10.1038/nature14539.

65. LI, Z. et al. A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. *IEEE Transactions on Neural Networks and Learning Systems*. 2022, vol. 33, no. 12, pp. 6999–7019. Available from DOI: 10.1109/TNNLS.2021.3084827.

66. MÜLLER, T. *Commissioning and validation of the underlying model of a test rig analyzing rotation and expansion performing assemblies in parabolic trough collector power plants*. 2017. Masters Thesis. RWTH Aachen Institute of Solar Research, German Aerospace Center (DLR).

67. WEISS, J. H. *Force Analysis on Flexible Pipe Connectors of Parabolic Trough Solar Collectors*. 2025. Masters Thesis. RWTH Aachen Institute of Solar Research, German Aerospace Center (DLR).

68. PIECOTRONICS, P. *Modell PCB-(M)320C52* [online]. [visited on 02/05/2025]. Available from: https://www.pcbpiezotronics.de/produkte/datenblatt/?untergruppe=vib_uni_durchgangsbohrung&h=PCB&m=320C52.

69. VIRTANEN, P. et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*. 2020, vol. 17, pp. 261–272. Available from DOI: 10.1038/s41592-019-0686-2.

70. MCKINNEY, W. Data Structures for Statistical Computing in Python. In: WALT, S. van der; MILLMAN, J. (eds.). *Proceedings of the 9th Python in Science Conference*. 2010, pp. 56–61. Available from DOI: 10.25080/Majora-92bf1922-00a.

71. TEAM, T. pandas development. *pandas-dev/pandas: Pandas*. Zenodo, 2020-02. Latest. Available from DOI: 10.5281/zenodo.3509134.

72. HARRIS, C. R. et al. Array programming with NumPy. *Nature*. 2020-09, vol. 585, no. 7825, pp. 357–362. Available from DOI: 10.1038/s41586-020-2649-2.

73. WANG, Z. et al. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*. 2004, vol. 13, no. 4, pp. 600–612. Available from DOI: 10.1109/TIP.2003.819861.

74. VAN DER WALT, S. et al. scikit-image: image processing in Python. *PeerJ*. 2014, vol. 2, e453.

75. PASZKE, A. et al. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. 2019. Available from arXiv: 1912.01703 [cs.LG].