Hamburg University of Applied Sciences

Faculty of Life Sciences

Pattern Recognition Algorithm for Signature Detection in Scatter Plots of Industrial Data

Master's Thesis

in Process Engineering

Timothy Essinger

[redacted]

Hamburg,

27.01.2022

Reviewer: Prof. Dr. Margret Bauer

Reviewer: Ph.D. Ashwin Venkat (Seeq Corporation)

This Master's Thesis was developed in corporation with Seeq Corporation.

Declaration of Originality

I confirm that the submitted thesis is original work and was written by me without further assistance. Appropriate credit has been given where reference has been made to the work of others. The thesis was not examined before, nor has it been published. The submitted electronic version of the thesis matches the printed version.

Location, Date: Signature:

Acknowledgment

I would like to express my deepest appreciation to Prof. Dr. Margret Bauer who supported me in all possible ways during and outside of the Master's Thesis. She encouraged and guided me on my journey and she has always an open ear. Furthermore, she opened my mind to think out of the box and she never runs out of creative approaches. Mrs. Bauer consistently shared her extensive experience in the field of automation and data science with me.

I would also like to extend my deepest gratitude to Ph.D. Ashwin Venkat for his support and for helping me to find the right scope of the work. Additionally, he taught me how to develop a software and how a software can add value for the customer. Mr. Venkat is always one step ahead and he helped to guide the project.

Many thanks to John Cox who always supported me during my time at Seeq. Mr. Cox helped me with all my questions on the Master's Thesis as well as in company-related topics. Furthermore, Mr. Cox helped me by testing the software and cutting the last corners.

I am extremely grateful to Prof. Ph.D. Nina F. Thornhill who always supported me with here nearly infinite amount of knowledge in all areas. Mrs. Thornhill encouraged, helped and guided me to find the perfect solution. It is a rare privilege to work with someone like her.

Additionally, my thanks goes to Alberto Rivas for teaching me important fundamentals in the field of software development. Mr. Rivas helped me at all times to develop and install the software and to detect and fix bugs.

And finally, I would like to thank my family who always supported me. Especially, I have to thank my mother and my wife. They always find the right words to cheer me up and they always have my back. I would have never made it this far and I will never forget their tremendous support. Thank you!

A few years ago, I started my journey to become an engineer. It was an interesting and instructive time, which is now coming to an end from an academic point of view.

Table of Content

Table of Figures

Tables

1 Introduction

Every plant in the process industry requires the ability to control important variables such as temperatures and flows during the production to ensure a safe manufacturing process. One possible way to monitor the performance of the control equipment is to investigate the data received from the process monitoring system and try to find characteristic patterns inside the data. In order to receive the process data from the operating system several softwares are needed. The first software achieves the goal of collecting and storing the data in the data historian system. The data are collected and stored with an operational data system of record. Currently, there are several software solutions available to store the process data of a plant. One of the commonly used programs is developed by OSIsoft.

At this point the data from the process are available and can be analysed. The current need in the process industry to analyse their process data with an easy-to-use software is fulfilled with the Seeq software. This software enables the customers to analyse their process data without programming knowledge in a highly visual environment. Before a high-level data analysis software such as the Seeq software was available, process engineers received their process data from the data historian system and developed their own applications to find insights in their data. These applications were, in most of the cases, only in one site available. Therefore, within a company the existents of several applications with the same purpose were common. With the open-source support, Seeq provides a solution for the above descripted problem. The Seeq software enables the opportunity to develop applications and shares these applications company-wide. One of the competitors of Seeq is TrendMiner which provides a similar data analysis software. At the moment, TrendMiner's software is not as sophisticated as Seeq's software and it offers a clumsier user experience. When it comes to data analysis in the process industry, the Seeq software should be the preferred choice.

One idea to decrease the workload for the process engineers is to automate the monitoring process and to order the found issues regarding to their magnitude. Therefore, the lower magnitude cases would be ranked low whereas the more

urgent problems are at the top of the list. Further, the high number of alarms are a challenge in the process industry (Hu, et al., 2017). By setting in a threshold for the minimum magnitude, only the cases above will cause an alarm which decreases the total number of alarms. In this work, the need of a full automated workflow for detecting issues in the control equipment and to rank them regarding to their magnitude is emphasized.

One example of control equipment are actuators which are field instruments that act on the process. Valves are a type of an actuator that can increase or decrease the amount of liquids and gases, for example to cool a reactor. Valves directly intervene in the process. With the nearly infinite possible use cases in the process industry, valves are one of the most common control apparatus. A good operation of the valves is therefore of utmost importance. To ensure an unobstructed manufacturing process it is of importance to monitor the performance of the valves. One of the most frequent problems that occurs in valves is valve stiction. In fact, valve stiction is not only the most common issue that occurs in valves, it is one of the most common issues in the process industry (in control loops) (Bauer, et al., 2016). Thus, developing a valve stiction detection algorithm would address an important issue in the process industry.

1.1 Problem Description

The aim of this work is to develop an algorithm that is able to use pattern recognition algorithms to detect valve stiction in scatterplots. This algorithm will be implemented as an add-on tool in the Seeq software. To use the algorithm properly the development of a user interface is required.

The fundamental idea is to extract characteristic information from the scatterplots to detect valve stiction in control valves. Further, scatterplots can be converted into images. Thus, extracting information from scatterplots can be interpreted as analysing shapes in images. Therefore, the topic points into the field of computer vision. Furthermore, it is required to minimize the amount of false positive results.

Another objective is to produce code that is not only understandable by software engineers. The reason is that an increasing number of engineers tend to develop their own algorithm and try to implement them. With a small amount of available

examples from the process industry, the implementation part can be difficult for non-software engineers. In order to provide a working example, which uses state of the art programming libraries, the source code of this work will be published. Additionally, the code has to be structured in a way that the user can install the add-on as a package. The chosen programming language to develop the package is python. The main reason for this choice is that python supports several libraries which makes it easy to analyse data and visualize the findings. For example, the pandas library is suitable for analysing large data sets as well as performing mathematical operations on the data. Furthermore, the syntax of python is easy to learn compared to other programming languages (C++, JavaScript).

For the user interface the user interaction should be set to a minimum without losing the ability to adapt the software for every use case. For the unexperienced user an additional explanatory information has to be provided within the interface. In addition, it is required to have a section to review the results. In addition, the data should be taken automatically taken from the Seeq software and it should be possible to send the results back into the Seeq software.

1.2 Structure of the Work

This work can be divided into three sections. These are the development of the valve stiction detection algorithm, the pattern recognition part and the implementation in python. The Figure 1 can be utilized for a first overview of the addressed topics. From a content perspective the intersection between the three circles describes this work.

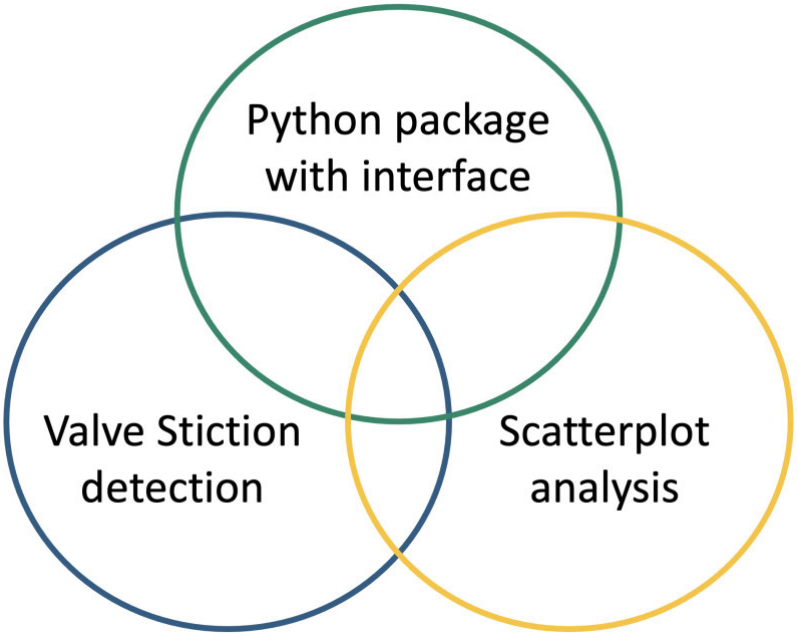


Figure 1: Venn diagram to show the addressed topics

For the implementation part it would not make sense to present every single line of code within this document. In the software industry the lifetime of certain lines of code is quite fast. One change in the function of a package or even the deprecation of a package can cause a restructuring of the entire code. As this document cannot be adjusted afterwards the implementation part is shifted into an open-source environment. Currently, the code is accessible in GitHub where it can be adapted according to the changes in the software industry. However, the concepts behind the program stays remain and these concepts will be presented within this work.

Before one is able to understand the key concept of stiction several foundations needed to be internalised. As a starting point, in Chapter 2.1, the feedback loop will be explained. Then valve stiction will be explained in Chapter 2.2. Followed by an overview of the existing stiction detection algorithms.

As mentioned, scatterplots can be transformed into images. Therefore, the algorithms to detect shapes in images are presented in Chapter 2.3.

The introduced concepts and algorithms are assembled in Chapter 3. Within this chapter the final concept of the algorithm is presented in detail. In addition, the user interface is introduced and all functionalities of the add-on are discussed. Further, the internal program structure will be discussed in Chapter 3.5.

How to develop a package can be taken from the code structure in the GitHub repository. To utilize the add-on in the Seeq software it is required to install the package which is located in the appendix of this document in Chapter 7.1. An up-to-date version of the installation guide can be found on GitHub as well as on the Python Package Index (PyPI).

The results are presented in Chapter 4 starting with the analysis of the validation data. Followed by the performance evaluation of the algorithm. At last, a conclusion will be drawn with an outlook into future projects.

2 Theoretical Background

The following chapter lays a theoretical foundation in which the key concepts are explained and illustrated. To understand valve stiction the concept of the feedback loop has to be clear. Therefore, the introduction of the feedback loop will be addressed first. The next topic is the physical explanation of stiction. In order to have an overview of the current stiction detection algorithms several approaches will be presented. Occasionally a part of the stiction detection workflow is an oscillation detection algorithm. Thus, two oscillation detection algorithms are presented. Stiction detection is helpful, but as outlined in the introduction, the measurement of the magnitude is of interest for this project. Therefore, the last topic for the stiction detection subject is an algorithm to measure the magnitude of the stiction.

As outlined in the introduction as well as in the problem description the scatterplots will be transformed into images. These images will then be further analysed with algorithms taken from the field of computer vision. Therefore, it is required to understand the used algorithms. They are presented in Chapter 2.3.

2.1 Feedback Loop

In the process industry it is importance to be able to directly change a certain parameter in the process. These parameters could be a certain temperature or pressure inside a heat exchanger or a conductivity sensor inside a reactor. In order to control these parameters feedback loops where developed. Feedback loops measure the current value and compare them with the desired value. One example of a feedback loop is illustrated in Figure 2.

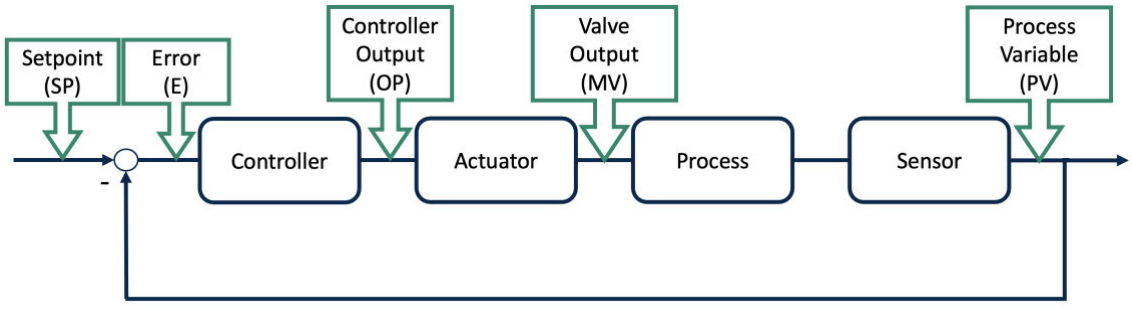


Figure 2: Feedback loop

The next component is the actuator which receives the output from the controller (OP) and act respectively. The actuators in this example valves and pumps. The valve would open the amount of percent coming from to the OP signal. This has a direct impact on the process. The next component measures the current value in the process and gives it back to the comparer (Ang, et al., 2005). The above explained workflow will start all over again. An example will help to understand the workflow of a simple feedback loop. First of all, the actuator and the process have to be defined. The process in this case contains a heat exchanger with the purpose to cool the product. The actuator is the control valve of the cooling fluid. The sensor measures the temperature of the product at the output of the heat exchanger. This example is shown in the following flow diagram.

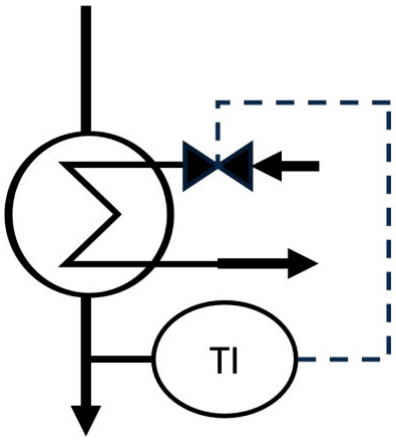


Figure 3: Example process

The following example should help to understand a simple control feedback loop. The controller in this example is a proportional controller. If for example, the current temperature in the product is 120 °C, the valve is 50 % open and the setpoint defined from the operator is 110 °C. Therefore, the absolute error value would be 10 °C. The controller would receive this value and change respectively the OP signal. The output of the controller could be to open the valve 5 % more which would lead to an total value of 55 %. This would lead to an increase in the mass flow of the cooling medium. Followed by a lower output temperature of the product. This loop will be continued until the setpoint is reached. One problem that could occur is a bad calibrated sensor leading to an unreliable output of the sensor. The actual temperature in the product could therefore only be determined if the sensor is well maintained.

2.2 Valve Stiction

Valves are the go-to-actuators in the process industry. Further, valve stiction is a well-known problem in the process industry (Brásio, et al., 2014) (Bauer, et al., 2016). In fact, valve stiction is the most common issue that occurs in valves. This is shown in Figure 4. In this figure a control loop is illustrated showing the most common issues that can occur in control loops in the process industry. Every issue is assigned to the related component in the control loop. The height of the issues reflects their occurrence. From the figure it can be determined that stiction is not only the most common valve issue. In fact, it is one of the most common problems in the process industry (Bauer, et al., 2016). This fact, once more outlines the importance of developing a stiction detection algorithm.

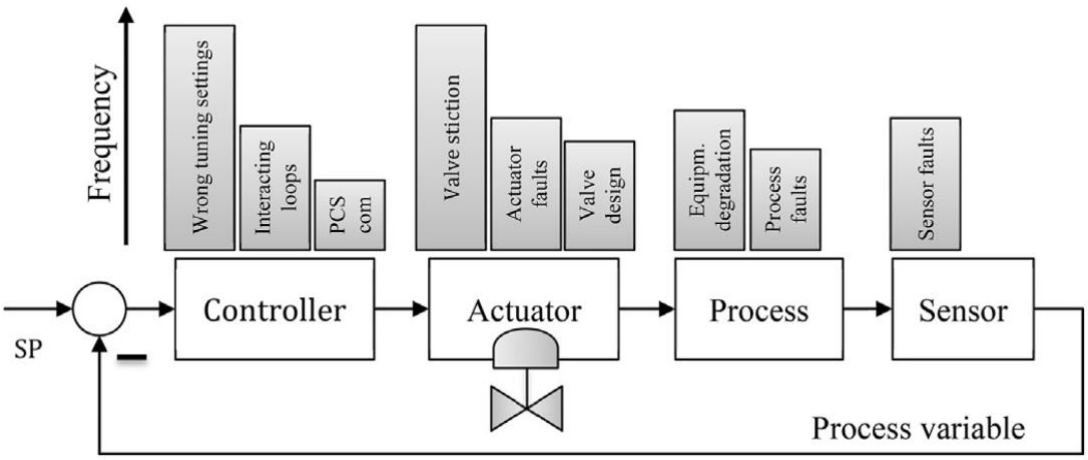


Figure 4: Most common issues in control loops (Bauer, et al., 2016)

Within the following section important terms, the physics behind stiction and several well-established approaches will be presented and explained.

2.2.1 Valves and Nonlinearities

First of all, a typical valve in the process industry will be explained. The Figure 5 shows the structure inside a valve.

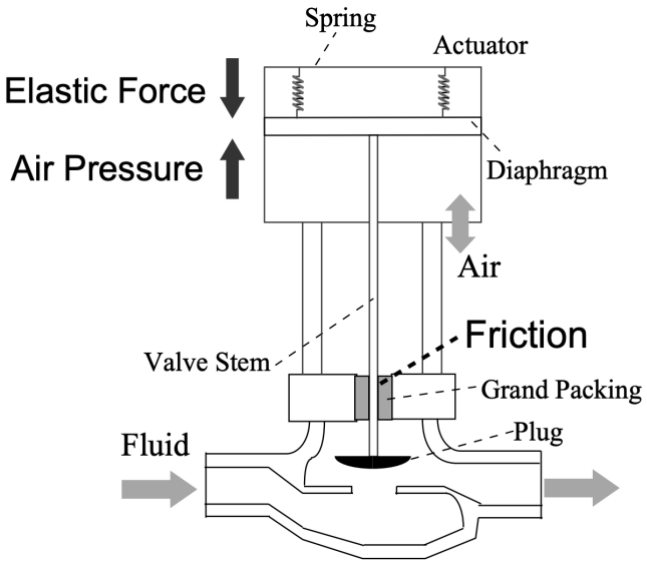


Figure 5: Structure inside a mechanical valve (Kano, et al., 2004)

Friction is the important term to focus on in Figure 5. Friction is per definition the force that acts against the motion of two connected surfaces (Kvam, 2009), (Fisher, 2005). Friction can be divided into the dynamic friction and the static friction. The dynamic friction is the force that is present when a certain object is under motion and in contact with other object or surfaces. A good example is a falling object surrounded by a gaseous fluid. In this situation the dynamic friction occurs because of the molecules in the fluid. These molecules have a different relative movement compare to the object and by connecting these friction occurs (Kano, et al., 2004). In addition, friction depends on the surface of the object. However, static friction is the force that have to overcome in order to move the object. Therefore, the valve stem has to overcome the friction between itself and the grand packing. All in all, when it comes to valve stiction, it is required to understand static friction.

The above explained forces can lead to nonlinearities in valve. These nonlinearities can be seen in the process data and have characteristic shapes. The Figure 6 gives an overview of the different scenarios that can occur. The plots are MV OP plots. The optimal case without nonlinearities would be represented by a straight line.

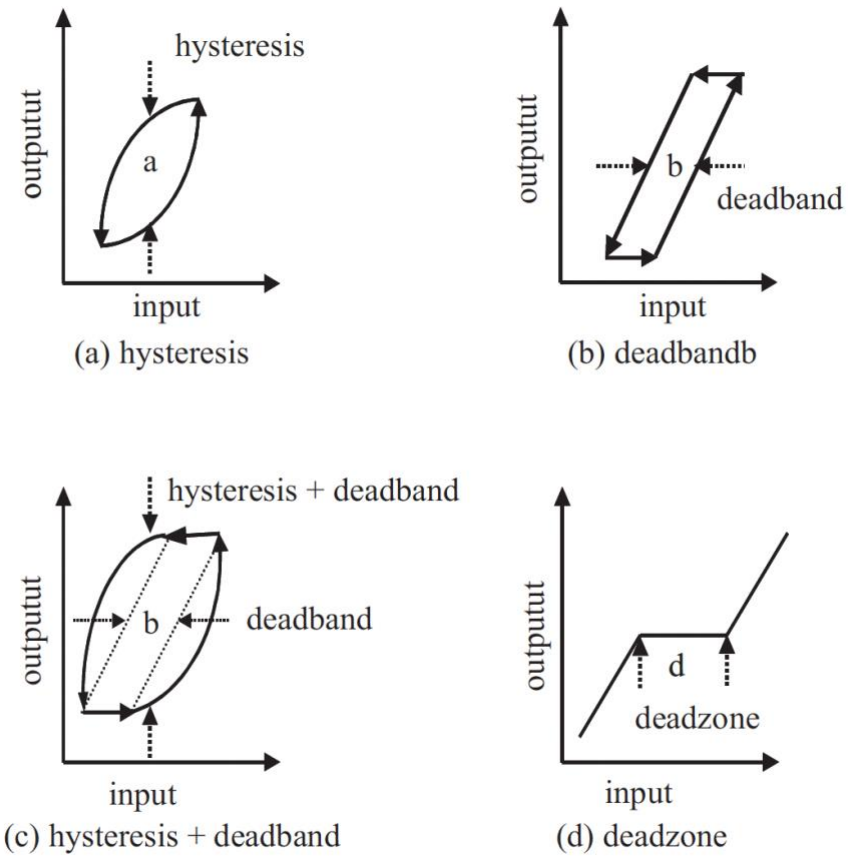


Figure 6: Overview of nonlinearities (Choudhury, et al., 2005)

Hysteresis describes the dependency of the output variable and their previous condition. Therefore, one input variable can have multiple output values depended on which path they are on. This is shown in (a). The situation in (b) where the input continues to change whereas the output stays remain is called deadband. Deadband is therefore the absence of a response to the input signal leading to a remaining output signal. The connection of the deadband and the hysteresis can be seen in (b). The deadzone is a zone where the output stays remain even though the input signal changes. The difference between the deadband and the deadzone is that the deadzone is independent of the direction in which the input signal changes (Brásio, et al., 2014) (Choudhury, et al., 2005) (Bacci di Capaci & Scali, 2018) (Choudhury, et al., 2007).

2.2.2 Valve Stiction in Scatterplots

For the explanation of valve stiction it is best to use the MV OP plot. At this point the terms MV and OP should be familiar otherwise Chapter 2.1 should be viewed.

The MV data are the direct output of the valve position and the data give direct information if the input lead to a change in the valve position. The MV OP plot is therefore just the comparison between the input of the valve and the output of the valve signal. In Figure 7 a MV OP plot is presented with all the characteristic behaviour that occurs with valve stiction (Choudhury, et al., 2005).

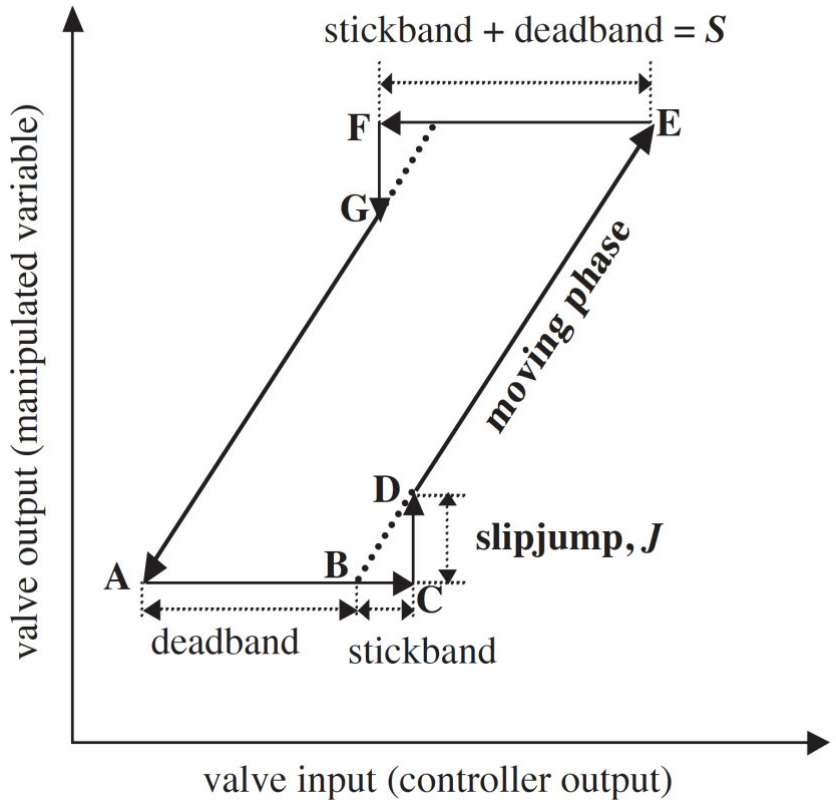


Figure 7: MV-OP plot of stiction signal (Brásio, et al., 2014) (Choudhury, et al., 2005)

Step by step explanation of valve stiction with Figure 7:

- Deadband: Once the valve receives the command to open or close, it will be impeded by the debris or other impurities in the valve. The valve gets stuck because it does not exceed the friction forces leading to a delayed response of the valve. Because of the not-responding valve, this characteristic is called deadband phase. This scenario is illustrated in point A to C (Choudhury, et al., 2005) (Brásio, et al., 2014).

- Slip Jump: The error however will further increase as well as the force to move the valve stamp until the valve starts to move. The stem starts to move at the moment where the moving force is strong enough to overcome the static friction force. Followed by the energy transformation from potential energy into kinetic energy. The source of the potential energy will

be taken from the actuator. It is common to observe a slip jump behaviour. The slip jump behaviour is also well described by “property of an element such that its smooth movement in response to a varying input is preceded by a sudden abrupt jump called the slip-jump” (Choudhury, et al., 2005). This is shown in the data between the points C to D.

- Moving phase: The valve continues to move until it gets stuck again. Followed by the same physical behaviour as above explained. This behaviour can be seen from point D to E (Choudhury, et al., 2005).

- Within the moving phase it could occur that the valve stem gets slower until it gets stuck again. The moving direction strays remain and only the stickband behaviour can occur (Choudhury, et al., 2005).

2.2.3 Oscillation Detection in Process Data

The first indication of stiction is an oscillation in the process variable (PV) and the controller output (OP). An oscillation is a periodical wave signal that has a regular behaviour. Therefore, it is best practice to search for oscillation behaviour in the OP and PV signal. The key is to detect the regular behaviour and differentiate between normal operating condition and oscillation. As this is a fundamental problem in the field of automation several scientists developed oscillation detection algorithms (Thornhill & Hägglund, 1997) (Thornhill, 2005).

2.2.3.1 Oscillation Detection with Zero Crossings

A widely used approach is to count the zero crossings of the error signal. The integrated absolute error will then be used to integrate the signal between the zero crossings. The cycles will be further analysed, in case this integral is above the threshold for the integrated absolute error. Further, the proposed algorithm evaluates if the amplitude of one cycle (data between two zero crossings) is above a user defined percentage. If this requirement is fulfilled the cycle will be counted. If the number of counted cycles exceeds the predefined number this time period contains an oscillating signal. The supervision time is the time frame

in which the counted cycles have to lay in. Otherwise, there is no oscillation in the investigated time period. The algorithm will then continue to integrate the zero crossings until the integrated absolute error is above the threshold. The sign change of the error signal enables the opportunity to determine the start and end time of the cycles. Further, it is required to store the time indexes of the zero crossings. In case, an oscillation is detected the zero crossing of the first cycle and the last zero crossing of the last cycle defines the start and end time of the oscillation. Therefore, the start and end time of the oscillation can be detected (Hägglund, 1995).

2.2.3.2 Oscillations Detection with the Integrated Absolute Error

In the paper a method to detect oscillation on- and off-line is presented. The two approaches mainly differ in the input parameters that needs to be given by the user. For the real-time oscillation detection, the controller integration time and the output range are needed (Thornhill & Hägglund, 1997).

The proposed methods used the integrated absolute controller error signal (IAE). The absolute error rate will be plotted in a graph and a threshold is used to classify if the investigated signal suffers from oscillation or not. The underlying concept is that oscillating signals tend to have a higher IAE value compared to a non-oscillating signal. The reason is that the data points between the zero crossing of the error signal will be integrated during the calculation of the IAE. Therefore, larger amplitudes will be taken into account. This IAE is shown in the Equation 1 (Thornhill & Hägglund, 1997).

$$IAE = \int_{t_{i-1}}^{t_i} |e(t)|\, dt \qquad \text{Equation 1}$$

In order to evaluate the performance of the algorithm, the authors used a performance index for the on-line detection. Further, an assessment index to evaluate the loop performance is presented whereas a lower value indicates a well-tuned loop. That is called a controlled loop performance assessment index (CLPA) (Thornhill & Hägglund, 1997).

Several problems that could occur in control loops are outlined in the article. To have a proper guidance the authors used power spectra graphs to differentiate between the issues for control loops. Additionally, the authors provided useful experience values to increase the effectiveness of the method. For example, should sigma r not be larger than one percent of the controller output. Or the values for eta = 0.15 are shown. These values where found out via the implementation of the algorithm in an industrial scale (Thornhill & Hägglund, 1997).

2.2.4 Valve Stiction Detection

Stiction can have negative effects on the process therefore it is important to detect it. However, stiction detection is not a fairly new topic in the process industry and over the last decades several scientists developed many approaches. In Figure 8 is an overview of the stiction detection methods presented. On a top level it can be derived that the detection methods are divided in manual and automatic methods. The focus of this work will be set on the automatic methods.

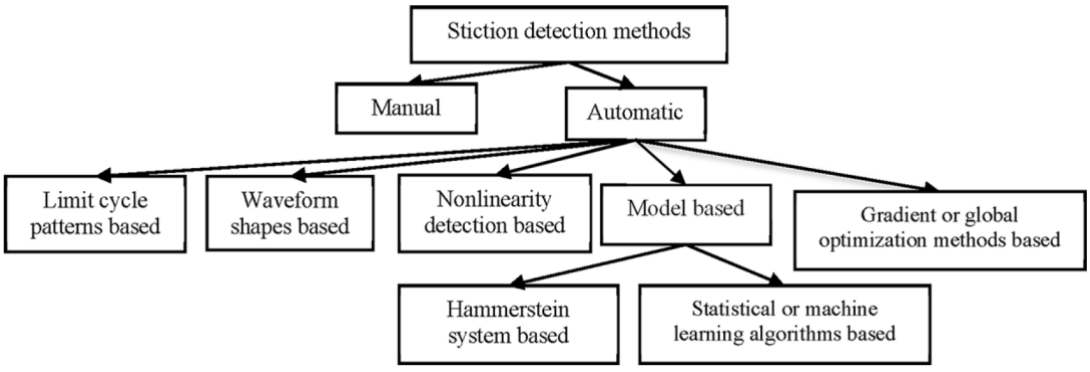


Figure 8: Overview of stiction detection methods (Zheng, et al., 2021)

The following section provides a brief overview of the overall idea of the different approaches. The explanation will be substantiated with at least one example for each approach taken from published articles. As this will be a high-level summary the requirement will not be to explain every approach in detail. It is more to

understand the overall idea and to lay fundamental understanding in how to detect stiction in control valves.

The first presented approach is called the limit cycle pattern-based approach. The limit cycle approach uses the characteristic shape of stiction in scatter plots. Dependent on the available data these plots can be PV-OP or MV-OP plots. Because of the two-time shifted oscillating signals the shape in the plots have an elliptical shape. The peculiarity is that these ellipses have sharp corners. A possible approach is to use pattern recognition algorithm to detect the shape in the plots. The investigated images could look like the plots on the right side in Figure 9.

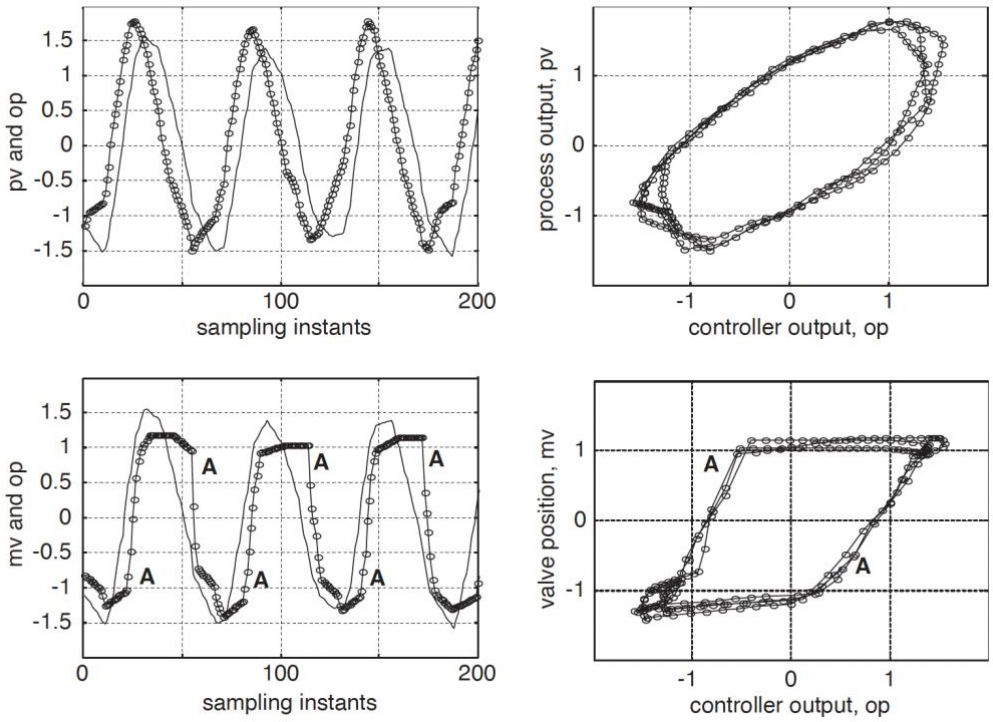


Figure 9: Stiction in different plots (Choudhury, et al., 2004)

The pattern recognition was performed with deep learning in the following paper. The authors tried to recognise the pattern in the images with a convolutional neural network (CNN). The structure of the CNN was taken from a contest for image recognition with deep learning algorithm. The use of CNN requires a high amount of training data which is in most cases not available. The high amount of training data can be reached with simulated data. Another CNN approach tries to manipulate the PV OP data in order to obtain a different shape. The goal is to achieve a butterfly shape that indicates the occurrence of valve stiction. These images are created via a simulation and are classified into three different

magnitudes of stiction (Zabiri, et al., 2020). Before the training of the model was performed the images were analysed by a round shape algorithm. The round shapes give information if the structure in the images has the form that the model needs. These images where used to train the CNN (Amiruddin, et al., 2019) (Vazquez, et al., 2019).

The next approach tries to detect the unique waveform of the OP signal. The waveform can be seen in Figure 9 on the left side. It is common to observe a square wave in the OP signal when a control valve suffers from stiction. Therefore, this approach tries to detect if the signal is closer to a square wave form, then to a sinusoidal. This can be performed by investigating single waves by calculating the similarity with a square and a sinusoidal wave (Hägglund, 2011).

As mentioned before, another characteristic of stiction are nonlinearities in the valve data. The result at the end is a nonlinearity index that evaluates the amount of nonlinearity. One example of evaluating the nonlinearity of the input signal is to compare it with the surrogate data of itself with an index (Thornhill, 2005).

The model approach tries to find a model which can reproduces the process data. The validation will be performed by taking the input data into the system. The output data from the model and these from the process will be compared. The model can fulfill the requirements once both outputs are similar. These approaches are divided into two subsections. The Hammerstein model and the statistical or machine learning approaches. The Hammerstein model consist of a static nonlinear part and a linear part. The static nonlinear part represents the valve, suffering from stiction whereas the linear part models the process dynamics (Bacci di Capaci & Scali, 2018) (Brásio, et al., 2014). An illustration of the Hammerstein model can be seen in Figure 10.

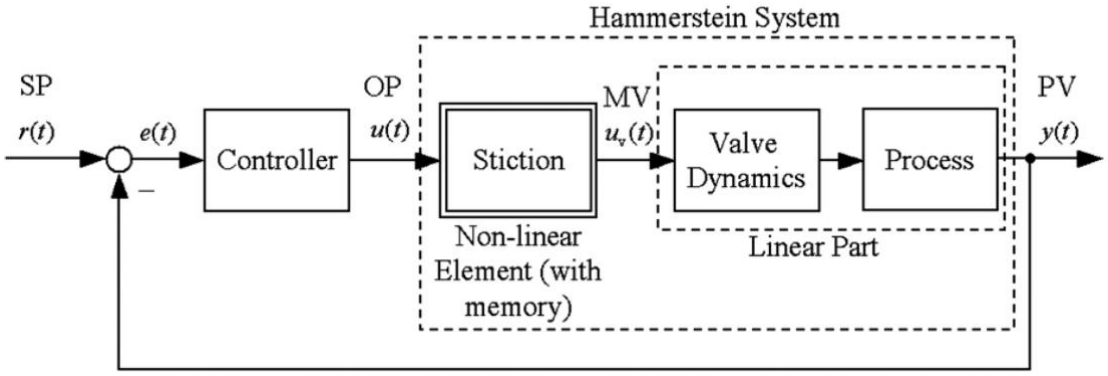
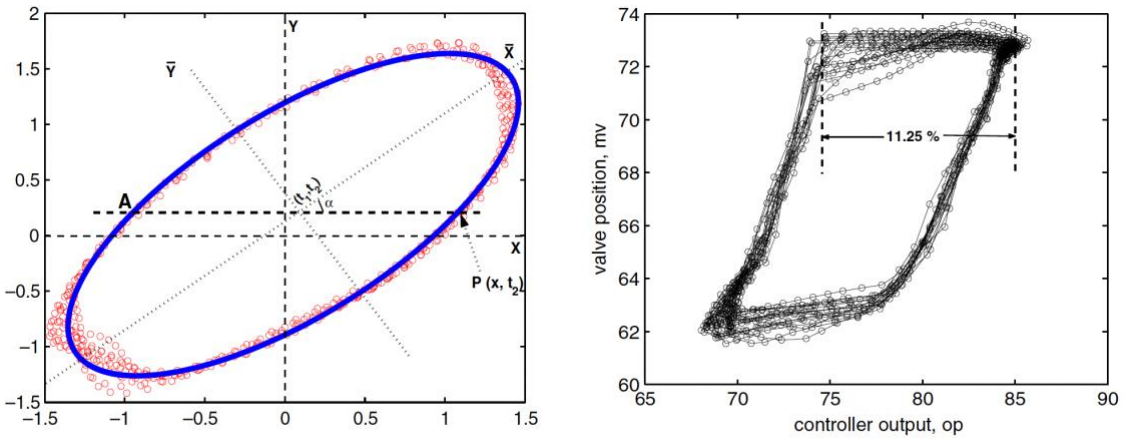


Figure 10: Hammerstein Model (Jelali, 2007)

The overall idea for the global optimization is to detect the cause of an oscillation from plantwide data. The reason for the oscillation can be stiction and therefore it can be used for detection. An example is the global optimization which is about finding the stiction parameters (Jelali, 2007).

2.2.5 Measure the Magnitude of Valve Stiction

In order to prioritise the previous detected stiction cases, an algorithm is needed to measure the magnitude of the stiction. As there are a large number of valves in every plant, a prioritisation of the detected stiction is needed to focus on the significant cases. The state-of-the-art unit to measure stiction is the percentage of the OP signal that stayed remain until the valve starts to move. This way of measurement is illustrated in Figure 11. The Figure 11 contains one example for the PV OP and one example for the MV OP plots.



(c) *pv-op* plot with fitted ellipse (d) *mv-op* plot

Figure 11: Measure valve stiction in plots (Choudhury, et al., 2007)

Measuring the width of the flat line in the MV OP plot leads to the magnitude of stiction that is present in percent. As most of the valves are not smart valves the MV data is not available. Therefore, another approach needs to be considered. The other approach is to measure the width of the ellipse in the PV OP plot. Hereby the process dynamics will be captured which leads to more unreliable results compare to the method in the MV OP plot. For this reason, the detected value is called “apparent stiction” (Choudhury, et al., 2007). The equation to calculate the width of the ellipse is given with the Equation 2.

$$AP = \Delta x = \frac{2\,m \cdot n}{\sqrt{(m^2 sin^2\alpha + n^2 cos^2\alpha)}} \qquad \text{Equation 2}$$

The variable m is the length of the major axis of the ellipse. Whereas n gives information about the minor axis. α is the angle of the rotation.

2.3 Shape Detection in Scatter Plots

Patter recognition gained over the last decade more and more attention. This trend is mostly based on the fast-increasing computational power combined with the lower prices of hardware components. These high computational resources are needed to go over all the single pixels in the investigated images. However, as the title of the thesis outlines, this work is about shape detection in scatterplots.

Therefore, a considerable amount of time will be taken by describing the utilized shape detection approach. A shape in this case describes a connected series of points which together represents an, by the human eye detectable, geometrical form. The previous definition outlines the crucial parts that have to be performed in order to detect a geometrical form in an image. As simple as this task is for the human eye as complex can it be for a computer. Several steps are needed to make this objective achievable for computers. These steps will be explained in detailed and different approaches will be presented in the following chapters.

2.3.1 Shapes Detection in Scatter Plots with Connected Pixels

The core idea of the algorithm is to detect the borders of the shape and define their relationship. A shape is detected by using the connected pixels approach which is a fundamental approach in the binary image analysis (Suzuki & Abe, 1985). The terminology has to be defined before looking for the details of the algorithm. A 1-pixel is a pixel contains structure information and could be a border point. A 0-pixel is defined as the pixel to fill the image or in other words the background or hole pixel. An outer boarder is defined as the border only surrounded by the frame. A hole border is defined as the inner boarder of a detected shape. The frame is defined by the uppermost row, the lowermost row, the leftmost column and the rightmost column. A pixel in the image will be called as (i,j) which should be read as (row, column). By looping over the image, the row variable i and the column variable j will increase from left to right and from top to bottom (Suzuki & Abe, 1985).

The Figure 12 can be used to understand the connected pixels approach. The approach investigates the directly connected neighbours and their pixel values. It can be further distinguished between investigating the four or the eight neighbours of the core pixel. For the 0-pixel neighbour connectivity the 8-neighbour will be taken. For the 1-pixel the 4-neighbour approach will be taken (Suzuki & Abe, 1985).

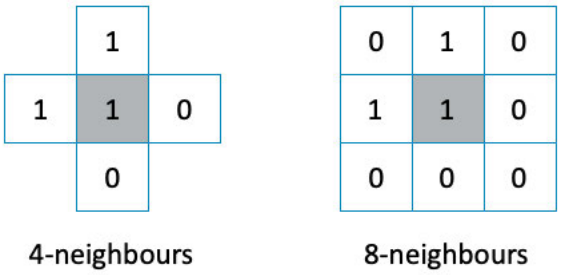


Figure 12: Example 8- and 4-neighbour connected components

In case the 1-pixel has a 0-pixel as one of its neighbours it will be labelled as a border point. To understand the relationship of the detected borders Figure 13 can be used. In Figure 13 is an image shown filled with three shapes. Additionally, the relationship to each other is presented. Every shape has to be closed to clearly define the borders. The amount of storage can be reduced by storing only the border information of the shapes in the image.

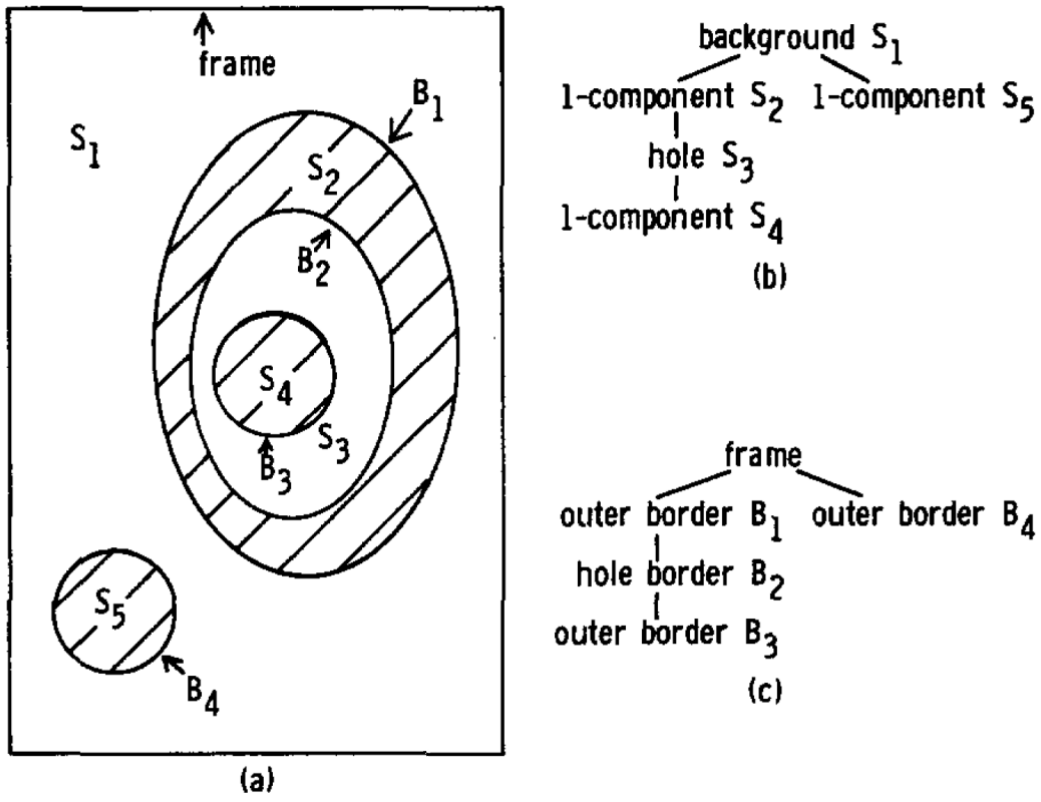


Figure 13: Relationship between borders (Suzuki & Abe, 1985)

Once a border is found the relationship can be determined by following the rules. The most important rules are explained in the following section. A detailed overview with all rules can be found in the source (Suzuki & Abe, 1985).

1. If the pixel after the current investigated one is zero, change the pixel name to – NBD (Suzuki & Abe, 1985).

2. Otherwise, change the pixel value to NBD but only if the current investigated pixel is not already on a followed border (Suzuki & Abe, 1985).

3. An outer boarder will be defined if the pixel value on the left, from the investigated pixel, is zero and if the current pixel value is one (Suzuki & Abe, 1985).

4. If the pixel, at the right side of the investigated pixel, is zero then it is called a hole border (Suzuki & Abe, 1985).

The last problem to solve is to decide if the detected border is a parent border. The relationship between borders can be taken from the Table 1.

Table 1: Relationship between detected borders (Suzuki & Abe, 1985)

Type of the border B' with the sequential number LNBD / Type of B	Outer border	Hole border
Outer border	The parent border of the border B'	The border B'
Hole border	The border B'	The parent border of the border B'

All the important rules are explained above. The workflow of the algorithm can now be explained in the following section. The presented steps can also be examined in Figure 14. The algorithm starts to iterate over the image from the top

left corner until it finds a 1-pixel. The loop ends by reaching the bottom right corner. The first 1-pixel point is stored as the beginning of a new border and it is unique labelled with the sequential number. Therefore, the pixel is now labelled as two. The reason for the start at two is that the frame of the image is always labelled with one. The next step is to continue the loop until the next pixel satisfies the required rules (Suzuki & Abe, 1985).

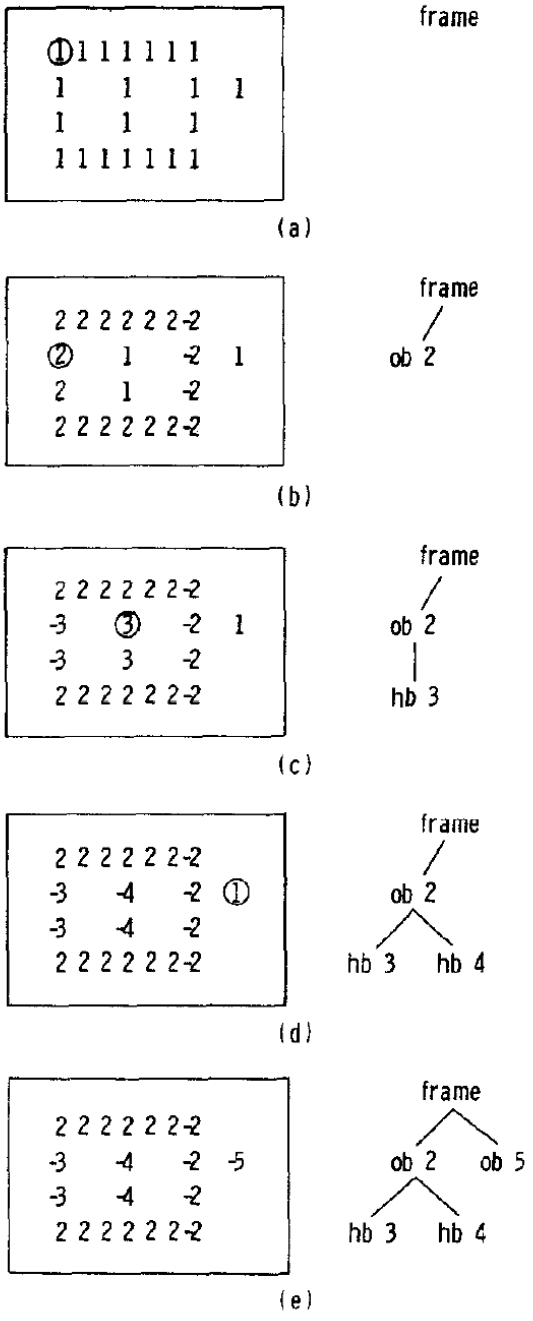


Figure 14: Example workflow of the border following algorithm (Suzuki & Abe, 1985)

In conclusion the input of the algorithm is the image filled with connected pixel areas. The results are labelled borders. These labels represent the relationship to the other borders.

2.3.2 Shape Comparison with Image Moments

In the section above an algorithm was explained to detect shapes in images which is able to understand their relationship represented by individual label numbers. As it would be of interest to compare these detected shapes the following section will present an algorithm for this task. The algorithm was developed in the 1962 from Ming-Kuei Hu from which it is called Hu-Moments. Image moments can be used to summarize the content in an image and it is defined by the Equation 3 (Hu, 1961).

$$M_{ij} = \sum_{x} \sum_{y} x^i y^j I(x, y) \qquad \text{Equation 3}$$

The indices i and j contain information about the order of the moment. Whereas the x and y variables are the location of the pixel on a two-dimensional plane beginning with the value one. The x variable provides information of the row and the y variable represents the column. Therefore, the top left corner pixel will be denoted as (1,1) and the following values will increase along the axis (Hu, 1961). This example is illustrated in Figure 15.

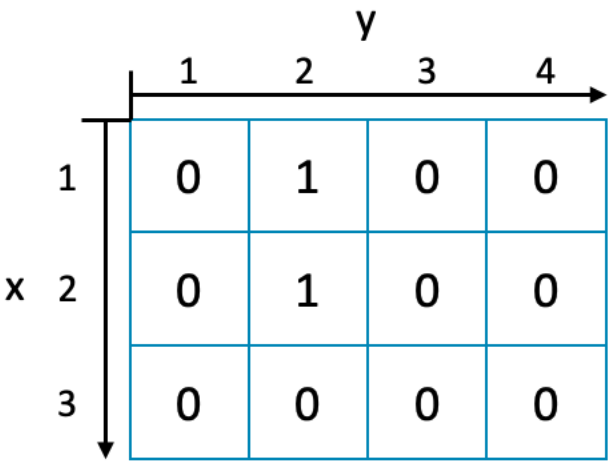


Figure 15: Example of an image matrix

When it comes to the pre-processing in computer vision there are two major image transformations named binary transformation and grey scaling. The two transformation are mainly used to store the structural information in a matrix. In a binary transformed image are all the pixel transformed to zero or one. These pixels, which are containing information about the shape in the image, will be labelled with a one in the matrix. The function I(x,y) from the Equation 3 will then be used to find all the values equal to one. The other method is denoted as grey scaling and transform every colour value into a value between zero and one. In this case the I(x,y) function will be used to find all values above 0. The function is not able to find directly these values. The values equal to zero will be multiplied as shown in Equation 3 with the other variables and end up with an total value of zero. Therefore, the current value in the matrix is represented by I(x,y) (Hu, 1961).

At this point all the variables and indexes in Equation 3 should be familiar. The next step is to show a clarifying example. For this example, the zero-order moment will be shown on Figure 15. For the zero-order moment the variables i and j are set to zero. Therefore, only the vales equal to one will be counted. At the end, the results are the sum of all the values equal to one. Or in other words the area of the shape in the image (Hu, 1961).

It can be seen that the location of the values in the matrix is of interest for the algorithm. In fact, this is one downside of the image moments equation. The goal is defined by an algorithm which is independent of the position, orientation or scale. In order to achieve the predefined goal, the Equation 3 has to be modified. The Equation 3 can be normalized by subtracting the mean of the variable x and

y from the current row and column value. The equation and can be found in the Equation 4.

$$\mu_{pq} = \sum_{x}\sum_{y}(x - \bar{x})^{p}(y - \bar{y})^{q} I(x, y) \qquad \text{Equation 4}$$

The mean of the row value will be calculated with the Equation 5.

$$\bar{x} = \frac{M_{10}}{M_{00}} \qquad \text{Equation 5}$$

Respectively the mean of the column is defined with the Equation 6.

$$\bar{y} = \frac{M_{01}}{M_{00}} \qquad \text{Equation 6}$$

Note the indexes of the moment equation. For the row values the sum of all values above zero will be calculated and multiplied by the current row number. The new equation is called centre moments which is nearly based on the fact that it finds the centre of the row and column with Equation 5 and Equation 6 (Hu, 1961).

The last step is to find the moments which are able to satisfy the required needs. The mathematical derivation to achieve this target would fill pages and would be out of the scope of this work. The in detail explained mathematical background can be found in the outlined paper (Hu, 1961). The important information is that Hu was able to define seven image moments which combined are able to satisfy the above defined requirements. In conclusion, the Hu moment algorithm is independent of their scale, rotation and position. If one is interested in comparing two shapes it can be done with the Hu moment algorithm.

2.3.3 The Structural Similarity Index

Another approach would be to investigate the similarity of the compared images. Similarity in this context is misleading and can be interpreted in different ways. The mean squared error (MSE) for example takes a closer look at the colour differences of each pixel. In many cases the information of the colour will be submitted vie the RGB values. These can have a magnitude from zero up to 255 each. Images with similar pixel colours and distributions will be marked with a lower MSE, compared to the images with a larger distance between the RGB values. A MSE value of one indicates full similarity and a value of zero means no similarity between the compared images. One of the biggest downsides is that the underlying structure of the images will not be given into the mathematical comparison. Further, the MSE contemplates the whole image at once. All in all, it can be conducted that the MSE seems to be outdated (Wang, et al., 2004) (Wang & Bovik, 2002).

Out of the fundamental idea of the MSE a new algorithm was developed. According to the opinion of the authors, the comparing of two images will end up in three characteristic values (Wang, et al., 2004). These are the luminance, the contrast and the structural values. In the following paragraph the mathematical formulars to calculate the above-mentioned characteristic values will be derived.

For the comparison of the luminance the mean intensity of the discrete data point will be used. To calculate the intensity, the colour values (mostly RGB values) are transformed into a mean intensity by the Equation 7 (Wang, et al., 2004).

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \text{Equation 7}$$

Therefore, the luminance can be defined with the Equation 8 (Wang, et al., 2004).

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{{\mu_x}^2 + {\mu_y}^2 + C_1} \qquad \text{Equation 8}$$

In order to compare the contrast, the standard deviation will be utilized with the Equation 9 (Wang, et al., 2004).

$$\sigma_x = \left(\frac{1}{N-1} \sum_{i=1}^{N} (x_i - \mu_x)^2 \right)^{\frac{1}{2}} \qquad \text{Equation 9}$$

The contrast will then be calculated by calculating the correlation between σ_x and σ_y with the Equation 10 (Wang, et al., 2004).

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{{\sigma_x}^2 + {\sigma_y}^2 + C_2} \qquad \text{Equation 10}$$

The constant C_2 can be defined as $K_1 << 1$ and L is the whole range of the pixel values. The constant C_2 and C_3 is defined by the exact same equation. The purpose of the constant is to decrease instabilities near to zero. The constant is defined with the Equation 11 (Wang, et al., 2004).

$$C_1 = (K_1 L)^2 \qquad \text{Equation 11}$$

Before the structure comparison can be performed, the images have to be normalized by dividing it by its specific standard deviation. Additionally, the luminance has to be identified from the images. The fundament of the formular is the well-known corelation function. Correlation is a commonly used method to determine if two datasets have a connection. Sometimes correlation will be mistaken by causality. It is important to know the difference between the two. Causality means that one of the datasets causes the value changes of the other dataset. However, correlation gives a value that indicates if the connection of the two signals will end up positive or negative. The correlation is defined by the Equation 12 (Wang, et al., 2004).

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x \sigma_y + C_3}$$ Equation 12

The similarity is the combination of the three calculated values. The single impact of the characteristic values can be adjusted by three parameters by taking values between zero and one. This is defined with Equation 13 (Wang, et al., 2004).

$$SSIM(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma}$$ Equation 13

The specific form of the SSIM when the parameters α β γ are the same is given in the Equation 14 (Wang, et al., 2004).

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x{}^2 + \mu_y{}^2 + C_1)(\sigma_x{}^2 + \sigma_y{}^2 + C_2)}$$ Equation 14

The careful observer will detect the constants (C) in most of the equations above. This constant simply takes care on the case that the sum of the compared mean or the standard deviation is close to zero. In fact, a value close to zero would lead to instable (Wang, et al., 2004) (Wang & Bovik, 2002).

2.3.4 Simplify Detected Shapes with the Ramer–Douglas–Peucker Algorithm

In order to simplify lines of shapes the following algorithm can be used. An introductory example is illustrated in Figure 16. Within certain error boundaries it can be seen that the black line can be represented by the blue line. Therefore, fewer points are needed to represent the same shape which is less storage and computational expensive.

Figure 16: Introductory example of line simplification

The workflow of the algorithm will be explained in the following section. The Figure 17 can be used to follow the steps.

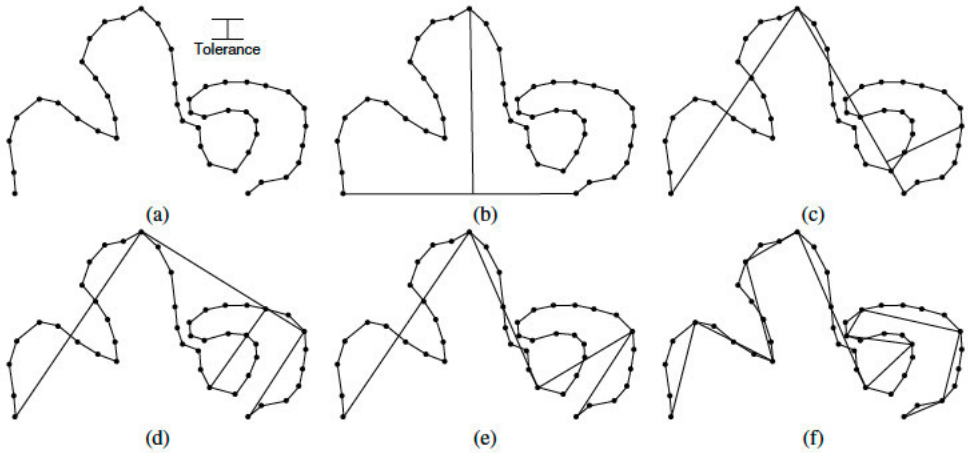


Figure 17: Douglas-Peucker Algorithm (Shin & Marquez, 2003)

The first step is the definition of the start and endpoint of the reducible line. The next step is to draw a line to connect these two points (b). Followed by defining the distance of all the points in between. The most distanced point from the line will be stored (b). From this new stored point two lines will be drawn to the two end points (c). Respectively the distance of all values between the two lines will be calculated (Douglas & Peucker, 1973) (Shin & Marquez, 2003).

The furthest point from the new defined line will be stored for the right line in (d). Then there will be two more lines drawn from the new defined line to the previous stored point and the end point (e). This procedure will be continued unless there is no point that is out of the tolerance boundaries (f). The same process will be applied to the left side of the geometrical form in (c). Connected with straight lines, the stored points represent the original geometrical form or line sufficiently (Shin & Marquez, 2003).

3 Methodology

Several algorithms were presented in the section above. In order to fulfill the given task these algorithms have to be connected in a proper way. As learned in Chapter 2.2, an automatic stiction detection algorithm requires an oscillation detection beforehand. In fact, detecting oscillations is a good starting point to ensure that the input signals of the stiction detection algorithm contain at least an issue with the valve. Otherwise, it could occur that the stiction detection algorithm detect an issue when there is none. The wrong detected stiction case would decrease the confidence in the results from the algorithm. Furthermore, it was outlined in the problem description that the utilized stiction detection algorithm uses a shape detection algorithm. Images are required for the shape detection and therefore they have to be created out of the plots. All in all, Figure 18 presents the workflow developed to address all the specifications.

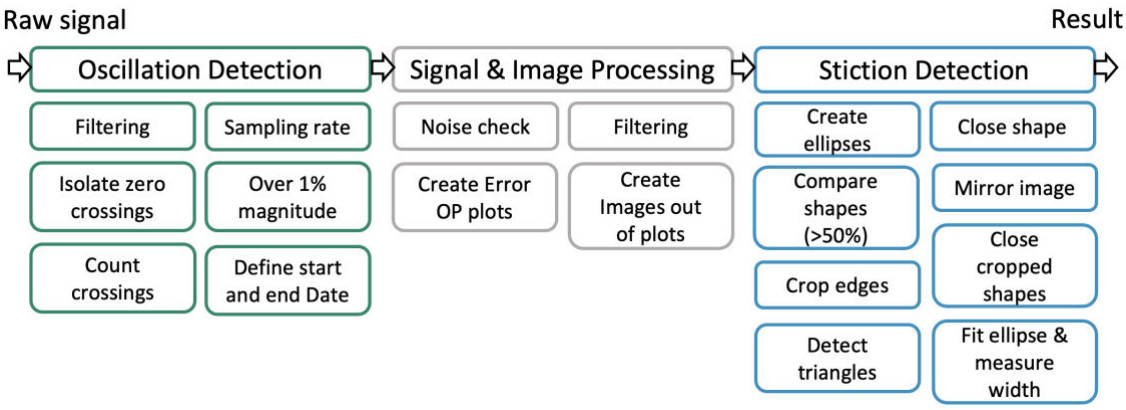


Figure 18: Workflow of the stiction detection algorithm

The workflow can be overwhelming at the first view. For this reason, all the mentioned points are addressed in the following sections. In general, it can be determined that first the oscillation will be detected and isolated. Followed by the signal and image pre-processing. Finally, the shapes in the images will be analysed regarding to their characteristics of stiction.

The detailed view will cover the underlying concept but will not show any code examples. This decision is mainly since the code could change over time whereas the underlying concept will stay remain. One reason for a change in the code could be that a utilized library adjusted their functions or even replaced one with

another name. Therefore, it is best to focus on the overall idea. In fact, an updated version of the code can be found under the open-source repository. After explaining the workflow in proper detail, the user interface will be presented.

3.1 Oscillation Detection Algorithm

The first step is to automatically detect oscillations in the raw signal. In addition, the start and end time of the oscillations have to be determined. The algorithm explained in the section 2.2.3.1 seems to fulfill all the requirements. This algorithm uses zero crossings to detect single cycles of the oscillation. Next it detects if the amplitude is above a certain percentage of a theoretical one. In case it is true, this cycle will be stored. If within the supervision time a certain amount of cycles will be reached, it will store the start and the end time as well as counting everything in between as one oscillation. The important point is that there are two layers of counting. The first is the time domain which is limited by the total supervision time. Additionally, an oscillation will be detected if the counter is above a certain number. Therefore, the supervision time is not the only counter and a fast oscillation can be better isolated.

As it is now explained which algorithm will be used, the detailed explanation of the oscillation detection workflow can be performed. The workflow can be viewed in Figure 19.

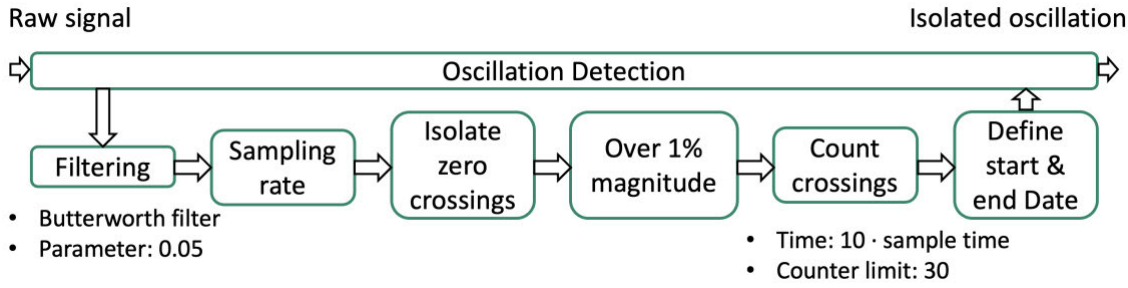


Figure 19: Detailed oscillation detection workflow

To detect zero crossings, a sign change of the error signal will be used. In order to improve the results of the algorithm it is therefore beneficial to filter the raw signal before. This step is important because of the high noisy signals in the process industry. To achieve this target a low pass filter named Butterworth filter

is used. An explanation of the functionality can be found in the literature source (Butterworth, 1930). After the signal is smoothed, the sampling rate will be determined. This is currently done by subtracting the time of two successive signal points. Once the second zero crossing is detected, it will be checked that the amplitude is over one percent. The one percent threshold was recommended by the authors. In case the amplitude of the investigated cycle is above one percent, the index of the first zero crossings are stored. If several numbers of cycles are above one percent (within the supervision time) an oscillation is detected. Then the index of the zero crossing of the last cycle is stored. The start and the end indexes of the zero crossings are now detected.

3.2 Signal and Image Processing

The output signal from the previous step is a list containing the start and the end index of the oscillations. The next step is to prepare the signals to create the images out of them. The workflow can be viewed in Figure 20.

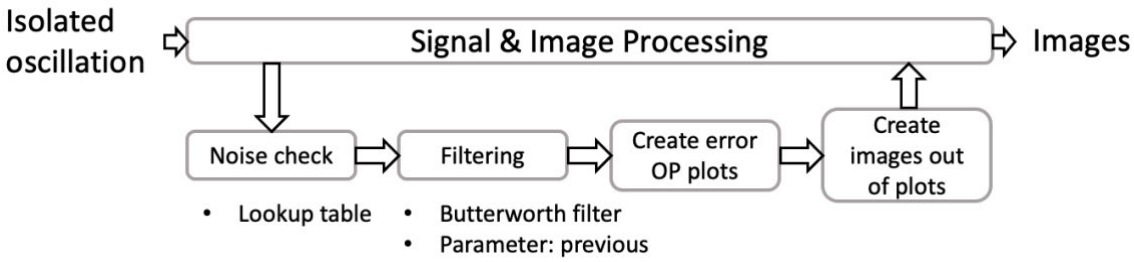


Figure 20: Pre-processing in detail

The first challenge is to process the high variety of input signals. In case the raw signal will be further analysed without pre-processing steps, the noise could destroy some of the important pattern in the data. Therefore, a smoothing of the signal is recommended. An excessive smoothing of the raw signal could destroy important patterns as well. All in all, some signals contain a high amount of noise and therefore the signals require a stronger smoothing. Whereas other signals could be taken directly to create the error OP plots. In order to address this issue an automatic noise detection has to be developed. This automatic noise detection should be able to select the filter parameter dependent on the magnitude of the severeness of the noise. The amount of noise was measured by the use of the

standard deviation and the mean. The raw signal will be taken and divided into several pieces which can be seen in Figure 21. This has to be done because the signals in the process industry changes highly over time. If the input signal is not divided into several pieces the mean can easily be biased by a changing setpoint. Withing the signals pieces the mean and standard deviation will be calculated and divided by each other.

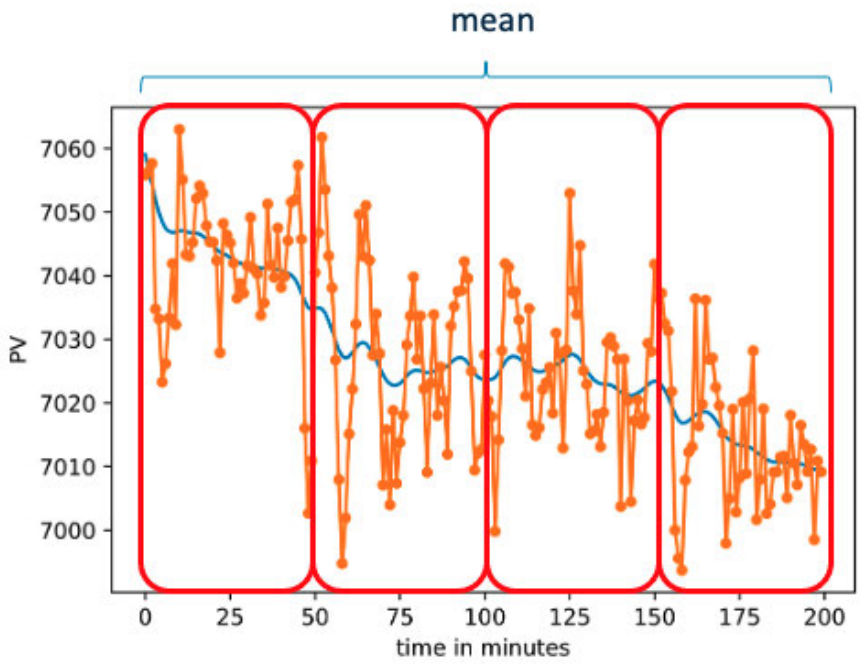


Figure 21: Noise index

After calculating the noise index a lookup table will be used to determine the filtering parameter. The lookup table was created by selecting a suitable filtering parameter by hand and plot it against the used noise index. The crucial part is to avoid an excessive smoothing of the raw signal which will lead to a loss of information. In fact, if the parameter is too high a stiction signal could be missed. A regression algorithm was then used to generate a function. This function will be used to select the filtering parameter automatically.

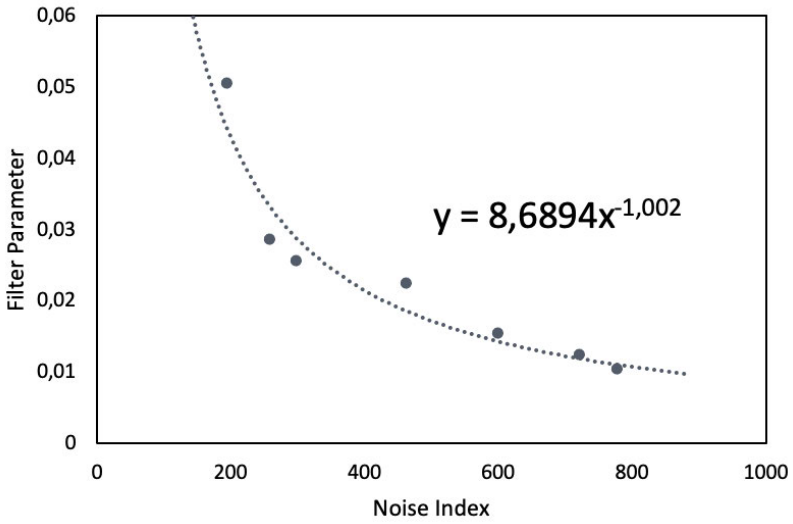


Figure 22: Lookup table for the filtering parameter

Once the filtering parameter is selected the raw signal will be filtered with the same Butterworth filter used in the oscillation detection. At this point the signal should be readable enough to create the error Op plot out of it. Followed by a rendering. This step is necessary to create an image out of the plot. The image is then a matrix contained with as many numbers as pixels. Further the image will not be stored locally and it is only used to generate the results.

3.3 Stiction Detection Algorithm

The core of the workflow is the stiction detection. The previous steps can be seen as pre-processing steps to reach this point in the workflow. As further explained several stiction detection algorithms are already developed. A selection of these were presented in Chapter 2.2.2. In the problem description was outlined that the focus of this work is about the pattern recognition in scatter plots. Therefore, all the non-shape-based approach can be sorted out. The fundamental idea behind this approach is that every issue in a scatter plot that can be detected with the sharp eye is worth to investigate.

In order to detect the characteristic shape of stiction a shape detector has to be used. The current approach is to use neural networks. These networks have to be trained with example images of stiction. In case the training was successful the network is able to detect similar shapes in other images. This field is not well researched and an over- and underfitting can easily occur. Additionally, the

required amount of training data is extensive, leading to the use of simulated data. The discussion if simulated data are sufficient to be used as training data will not be discussed as this point. At last, the use of neural networks demands a high expertise in this field and it is a highly complex method. Leading to the conclusion that neural networks are not the best choice.

From the neural network was learned that the used approach should be at first easy to understand. Further it should use a less amount of training data and the used parameters should be understandable. As none of the available algorithm fulfill these requirements sufficiently a new algorithm has to be developed within this work. This workflow can be seen in Figure 23 and will be further explained.

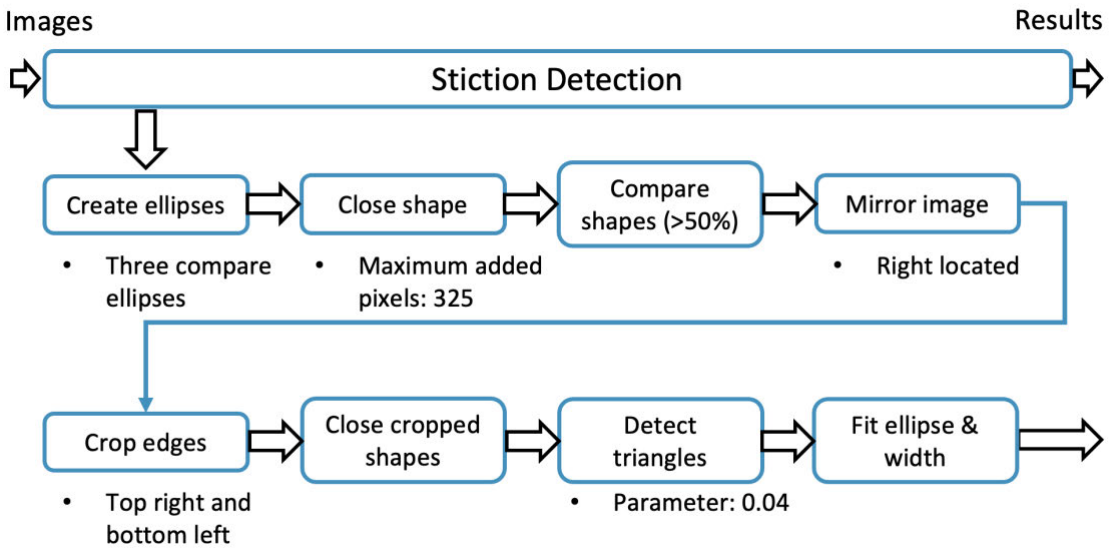


Figure 23: Stiction detection algorithm workflow

The first step is therefore to detect a shape in an image without the use of neural networks. The idea behind the developed approach is to first detect the shape in an image. The next step would be to compare the detected shape with one of a database. In case the similarity is greater than a threshold the shape in the investigated image shows similar characteristics. The initial task to process is to detect a shape in the image. This can be done by the connected component approach. In fact, a well-suited approach was already presented in Chapter 2.3.1.The output of the algorithm are boundaries of the shape stored in a matrix. A prioritization has to be done if more than one connected structure can be found in an image. Currently, the largest connected structure will be taken for the further analysis. To detect the shapes, they have to be closed beforehand. This is done

by the use of the parent child relationship explained in Chapter 2.3.1. In case the shape is open, a black pixel will be added and the algorithm will check again. Care must be taken to not change the structure too much. Therefore, a maximum number of added pixels will be set.

At this point the shape in the image is detected. The next step is to compare it with one shape from the database. This can be performed with the image moments algorithm presented in Chapter 2.3.1. The compared images will not be stored locally nor taken from a server. This approach requires a very limited amount of compared images. The images will be created every time the algorithm is called. A database would be exaggerated for this amount of files. While analysing, the test data form the SACAC database, it was found out that there are mainly three kinds of ellipses. These differ mainly in their width. Thus, three different ellipses will be used for the comparison which are presented in Figure 24.

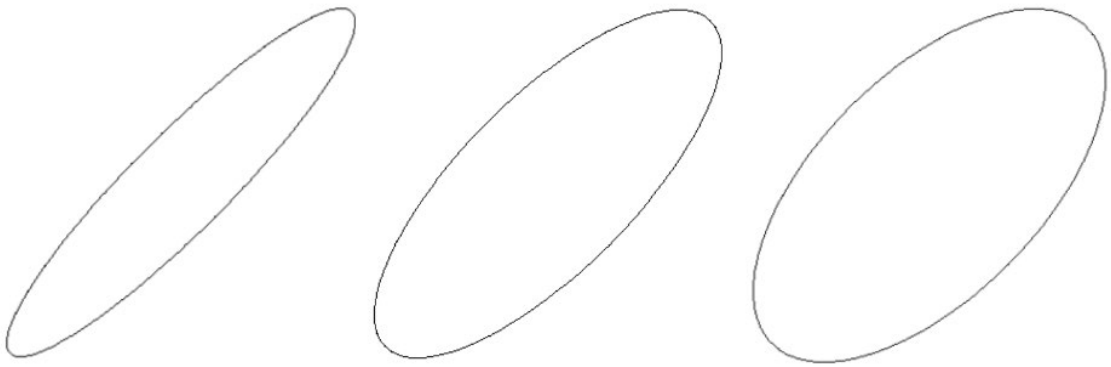


Figure 24: The ellipses used for comparison

Considerate Figure 25 to understand the next step. The Figure 25 contains two ellipses. One contains the characteristics of stiction whereas the other does not. In Chapter 2.1 was explained why the sharp cornered ellipse shows the characteristic behaviour of stiction. The round cornered ellipse shows only an oscillating time shifted signal.

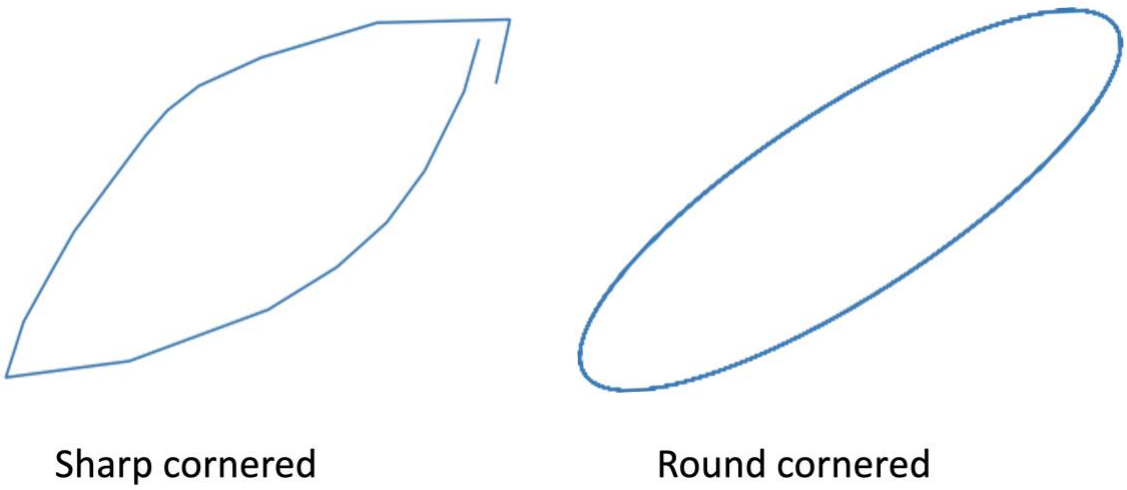


Figure 25: Round and sharp cornered ellipses

If there is an elliptical structure in the image it will be detected as this point. The next step is to focus on the sharp cornered edges. This can be done by cropping out the sharp cornered edges from the ellipse. By closing the two end points of the ellipse a new shape is created. An illustration can be viewed in Figure 26.

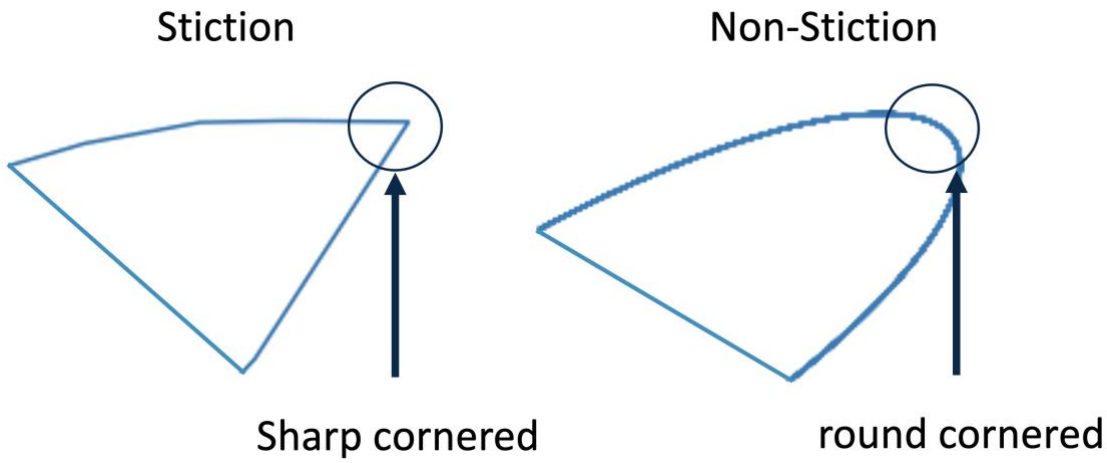


Figure 26: Cropped edges of the round and sharp cornered ellipses

By comparing the two shapes one can derive that the ellipse with the sharp corners looks like a triangle. In fact, if one is looking for stiction, the shape to look for is a triangle in the cropped edges of the ellipse. The triangle can be detected with the algorithm presented in Chapter 2.3.1. The presented algorithm is able to reduce the number of points to the bare minimum. In fact, if a shape only needs three points to be shown, it is triangle. Of course, this algorithm requires a parameter tuning. Within certain boundaries the two shapes in Figure 26 can both

be detected as triangles. This problem was solved experimentally. The found value can differentiate between a triangle and a square. In conclusion the left shape will be detected as triangle and the right shape will be counted as a square.

In case a shape does fulfill all these requirements it is very likely to be stiction. This conclusion can be derived under consideration of all the gates the signal has to pass. First the oscillation detection. Followed by the ellipse detection and at last the detection of the sharp corners. The accuracy will be investigated in Chapter 4.2.

3.4 User Interface

A user interface has to be developed in order to provide a convenient user experience. Before this topic can be explained, the requirement on the interface has to be defined. As the algorithm should work in corporation with the Seeq software there should be a section to select the signals from a Workbench. In the Seeq software a Workbench is an interface where the data from the database can be visualized and calculations can be done. The main data analysis part is taken place at this point. Additionally, the results of the analysis have to be visualized and shown to the user. The opportunity to send the results back to the Seeq software is another requirement. An explanation should be provided for the users with limited experience in the stiction analysis. All these requirements will be explained within the following section. An overview of the user interface can be seen in Figure 27.

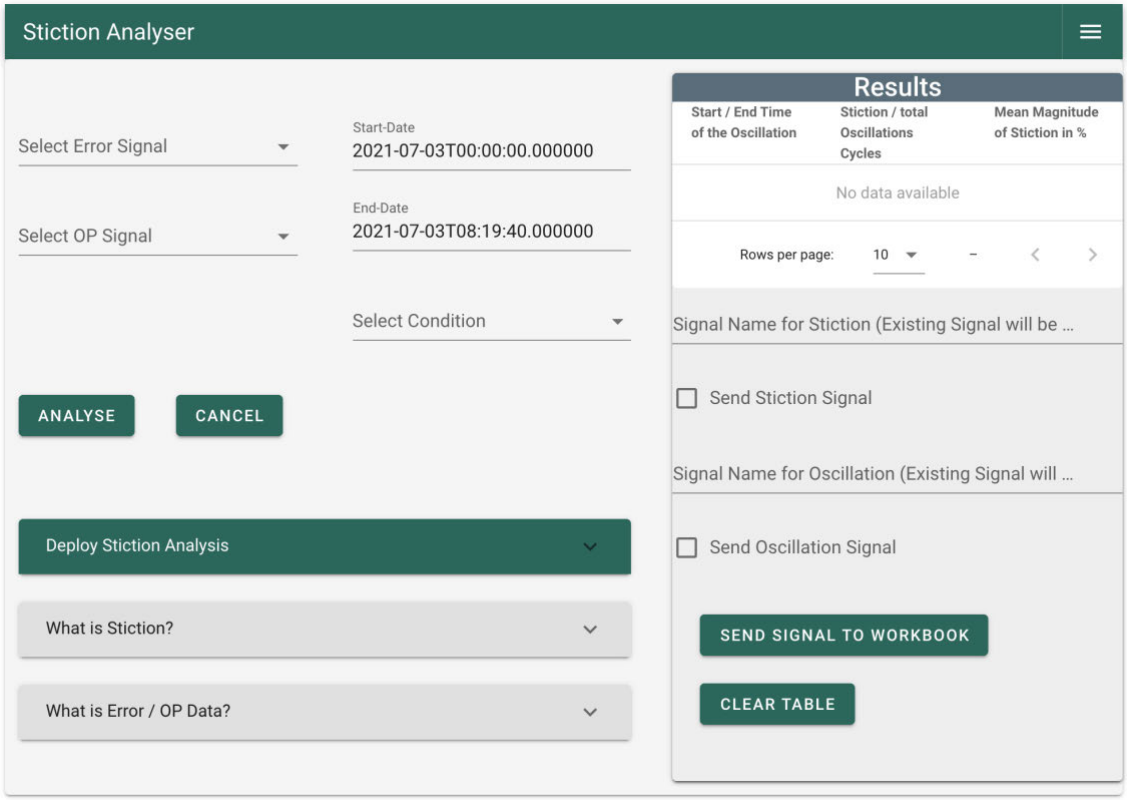


Figure 27: User interface

As this can be overwhelming at the first view, every section will be explained within the following section. First of all, the signal selection and the start and end date will be explained.

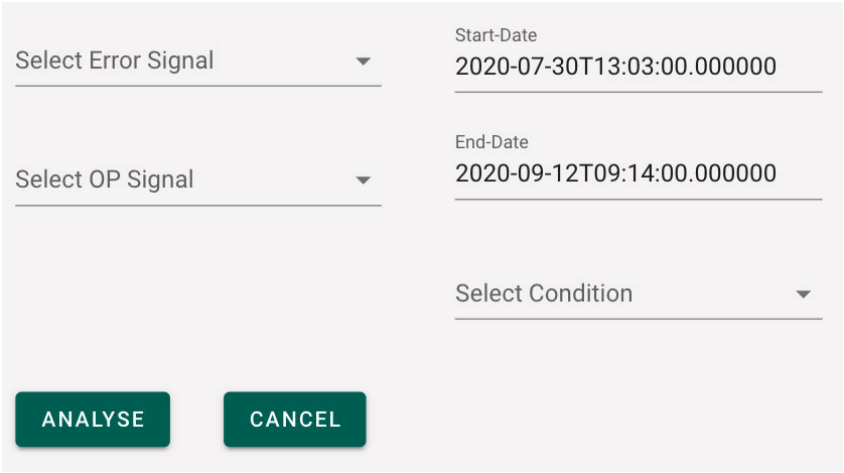


Figure 28: Selection of the signal and the condition in detail

The error and the OP signal from the Workbench can be selected in the outlined section. This section can be seen in Figure 28. Until the beginning the start and end date are already selected with the display range taken from the Workbench.

If one is interested in extending this time it can be done by adding the required start and end date. Below this section the user can select a condition. This can be useful in a variety of cases. For example, if there is a downtime in the plant these data can be sorted out.

The next section is about showing the results of the algorithm provided with an example. This section is illustrated in Figure 29. The start and end time of the oscillation is given in the first column. The next column gives information about how many cycles contain a sharp cornered ellipse out of the total oscillation cycles. At last, the mean magnitude of the stiction will be given. Note that this percentage is the apparent stiction magnitude as explained in Chapter 2.2.5. Within the calculation a signal for the stiction will be created as well as a signal for the oscillation detection. The user can now write the name of the signals and can push them back into the Workbench.

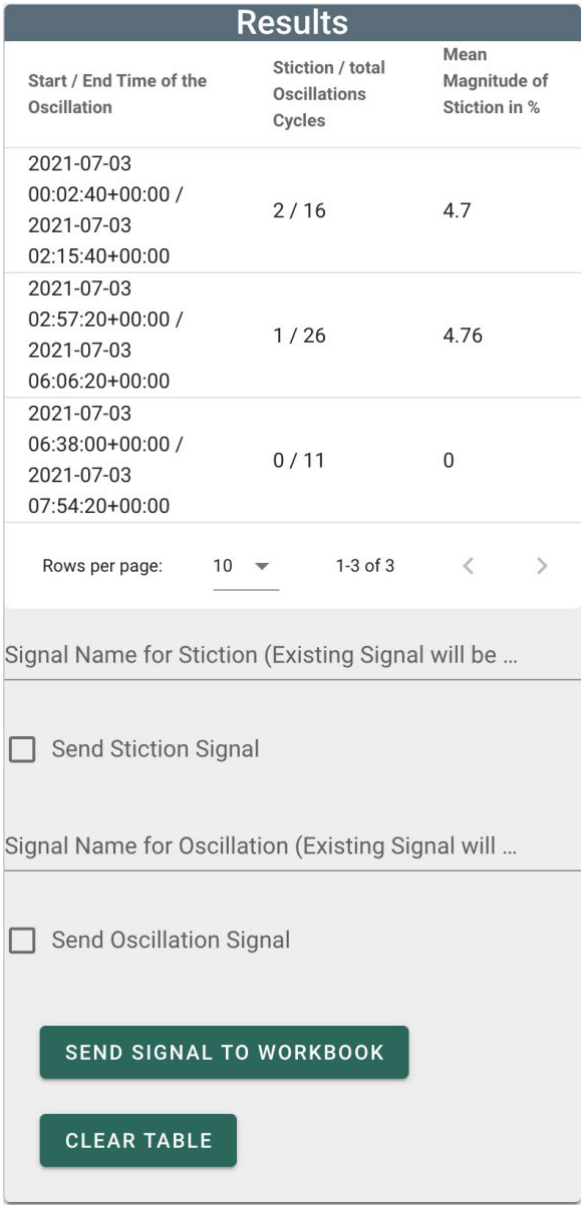


Figure 29: Results section in detail

The deployment button will be used to schedule the stiction analysis. In this context scheduling means that the analysis will be automatically executed. The user has to select the name and how frequency the code should be executed. The start date is set by the display range and the end date is also given by the display range. Therefore, the analysis will analyse the same time range. For example, the displayed range is about seven days, and the frequency is one time per month. The code will then be executed every month after the first seven days. This is shown in Figure 30.

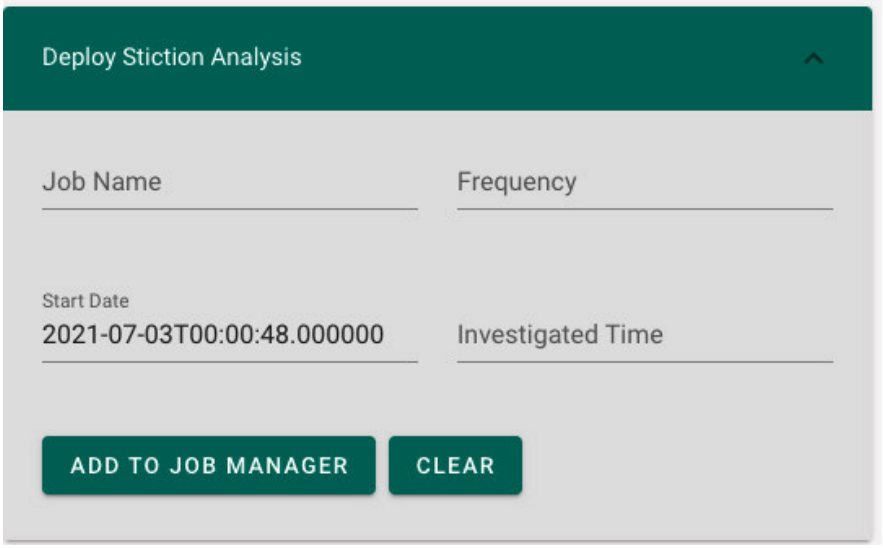


Figure 30: Deployment of the stiction analysis in detail

3.5 Internal Program Structure

In the previous chapter was explained how the algorithm works on a wider perspective. In order to understand the workflow in detail the inner structure of the program will be explained. Viewing the code structure from a top level it can be detected that there are two main classes from which the needed functions are called. One class contains all the functions that are needed for the signal pre-processing. The other class collects functions that are needed for the computer vision part. In addition, there are several support functions. To increase the maintainability of the code these classes and function are stored in different .py files. At last, there is a main file in which the user interface is stored. Within this file the main loop over the provided data is performed. It would lead to confusion by putting all the functions and classes in one file. Therefore, several support functions are called in the main file. When it comes to bringing the pieces together there are two important functions. One is for creating the plots and the images. The other is for detecting the shapes in the images, checking if these shapes are similar to an ellipse and measure the magnitude of the stiction. In fact, the main file is only a pointer to the functions where the calculations will be performed. The results of these calculations are stored in lists or arrays. The structure of the code can be seen in Figure 31.

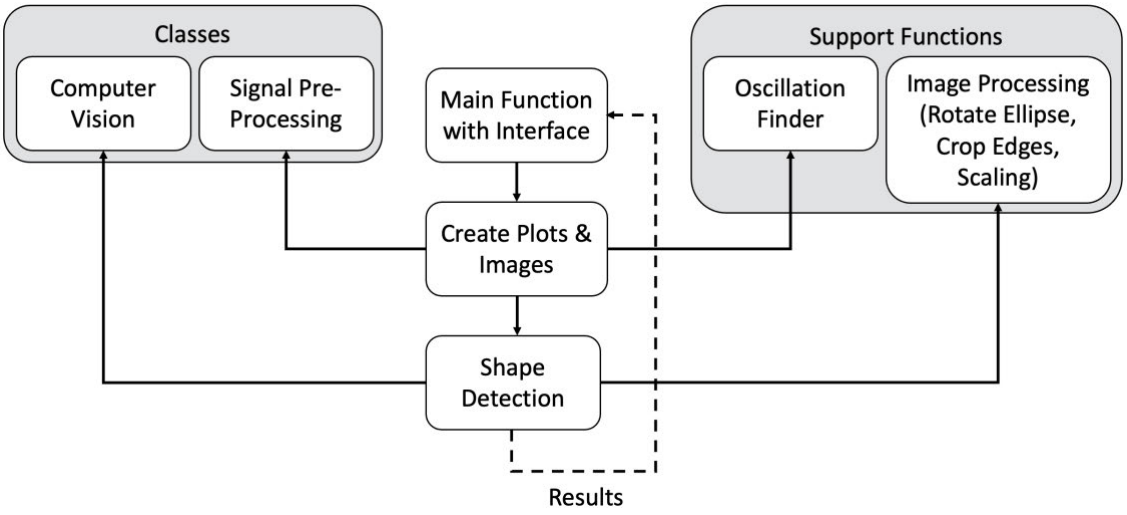


Figure 31: Program structure – Classes and functions

In the section above the programming structure was clarified. The next step is to understand the internal data structures and the programming techniques. Within the following section a deeper look inside the data structures will be taken. The Figure 32 can be investigated for a first introduction.

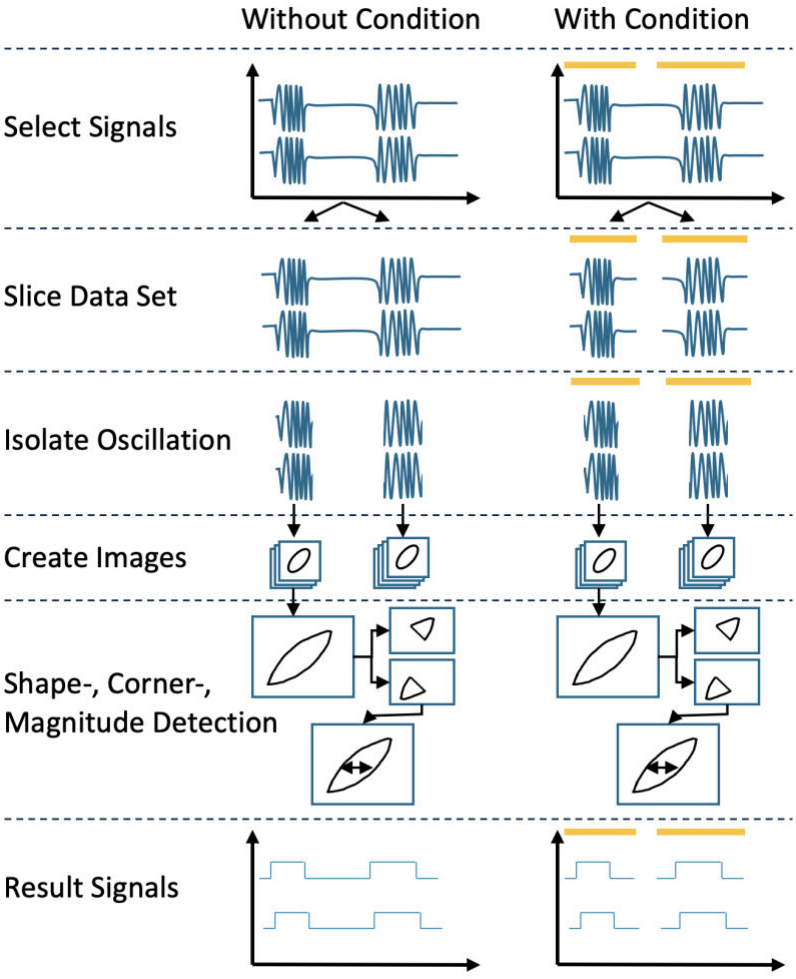


Figure 32: Workflow of the program without and with condition

By starting the add-on, a pull request will be sent to the server. This step is performed to receive the signals and the displayed conditions in the Workbook. The program starts once the user is satisfied with the input and clicks on “ANALYSE”.

The following section is difficult to understand without an illustration of the workflow. Thus, the workflow of the program can be seen in Figure 33.

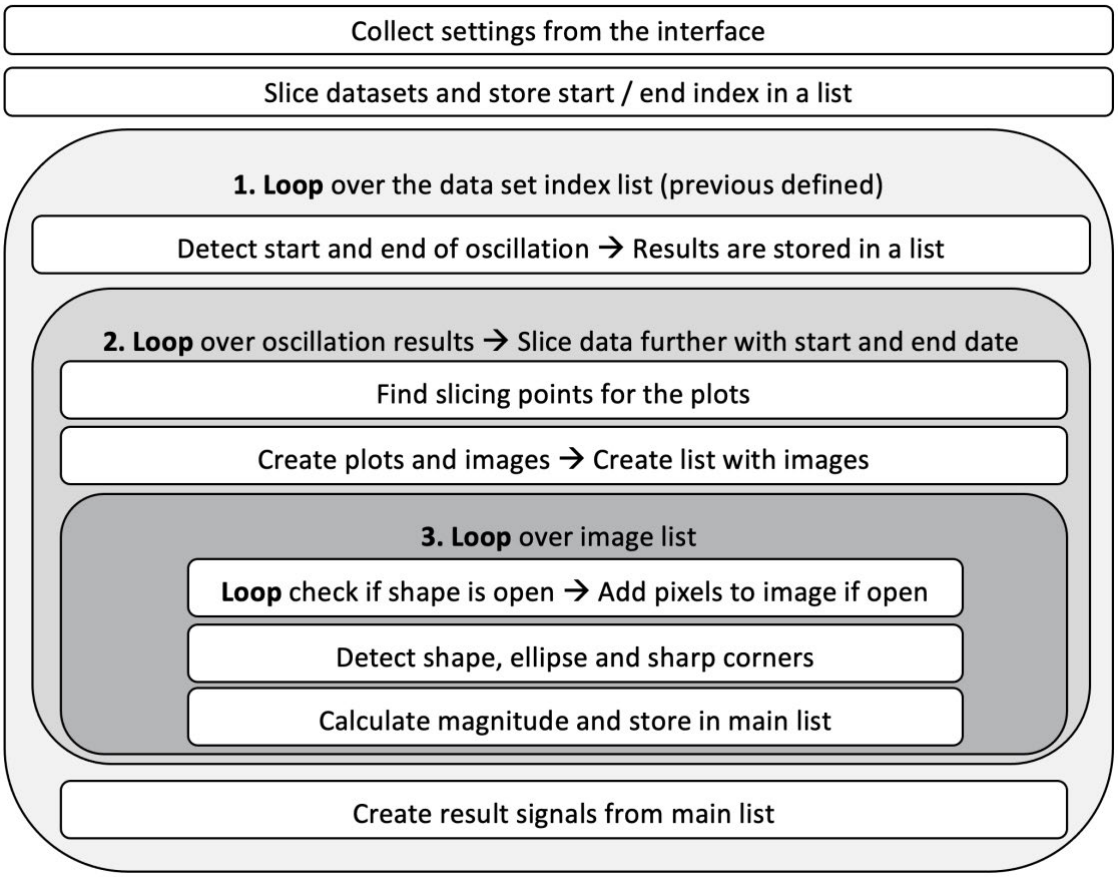


Figure 33: Program structure - Loops and data structure

At first the user inserted information will be collected and stored. In case the user selected a different time range (then the display range), a new pull request will be performed with the start and end time provided by the user. At this point the investigated data are selected and pulled. There are two possibilities that have to be addressed. In the first case the user does not select a condition. Therefore, the whole data set can be taken without dividing it. In some cases, it can be beneficial to split up the data set and only investigate certain areas within the time range. A well illustrating example is process data containing turnaround information. Sometimes these downtime data may not be of interest and a condition could be used to skip this time period. Therefore, only the data within

the condition will be investigated which leads to a divided data set. An example is provided in Figure 32. The slicing indexes will be determined and stored in a list. This list is fundamental for the first loop. In this loop the data will be sliced at the stored points. The large data set will be sliced at this point. In case the user does not specify any condition there will only be two indexes in the list. These are the start and the end index of the unsliced data set. The next step is to loop over the list and to slice the original data set at the specified indexes. The sliced data set will then be further analysed. It is less intensive for the memory to work only on the sliced list.

At first an oscillation detection algorithm tries to determine if the subsample contains an oscillation. If an oscillation is present, the start and end date of the oscillation will be detected. Further, consecutive oscillation periods are detected as a connected oscillation period. The reason for establishing this is that oscillations can be longer than the investigated time range and should be labelled as such. Within the condition several time ranges could be detected. These time ranges are stored with their start and end index in a list. The next loop receives this list and slices the data set further. In fact, within a condition several unconnected oscillations can occur. An example will help to understand the importance of analysing them separately. In case the condition contains signals from several days and the oscillations are analysed all together. In case there are two stiction cases within the analysed time range. Further are these cases not connected. The calculation of the mean magnitude of stiction would not be reliable on the single stiction cases. Therefore, the cases will be calculated separately.

However, the next step is to slice the signal after every wave. This step is required to create the plots and the images. The images for every cycle will be stored in a list. The list will be cleared after the analysis. Thus, the calculation does not require an interaction with the local storage of the user. The created list contains all images in order. The next loop goes over the images and tries to detect an elliptical shape as well as sharp corners. The last step is to calculate the magnitude of the stiction and to store all of the results in a list. This list will be appended in every iteration. Once all the images are analysed and the results are stored, the Loop 2 will continue with the next oscillation period. Then, all the oscillation periods in this condition are analysed the Loop 1 will go to the next

condition. If it does not exist, the analysis is completed and the results will be displayed in the results section on the screen.

If the user is satisfied with the results, they can be pushed back into Seeq. The result signals have to be created to achieve this target. There are two signals that the user can push back into Seeq. The first one is the stiction signal containing the information at which point stiction is present in the process data. The value of the signal is the apparent amount of stiction at every time step. It is important to understand that the mean apparent amount of stiction will be calculated for every single oscillation period. Therefore, the apparent amount of stiction can increase or decrease within the analysed time range. The other signal is the oscillation signal. This signal gives information at which time period the signal is oscillating. An oscillation is denoted with a 1 whereas no oscillation is labelled as 0. By selecting a capsule for the analysis, the analysis will only investigate during this time period. This is reflected in the result signal where only a value will be stored within the time range of the condition. If no condition is selected the result signals will be created for the entire time range. The result signals are illustrated in the bottom of Figure 32.

4 Results

Withing the following section the previous presented workflow will be tested with several industry use cases. At first the testing data will be shown and analysed to prove that these are well suited to test a stiction detection algorithm. At the moment there is no algorithm which is able to detect everything without limitations. Sometimes these limitations were set before the development of the approach. Limiting an approach can be useful to ensure the detection of a special behaviour. Therefore, the aim of this algorithm is to get a minimum number of false positive results. The Figure 34 helps to understand the different scenarios. This is also called the confusion matrix. In this context true positive means that the algorithm predicts stiction and it is actually there. A false positive is the opposite of this meaning stiction was detected but there is none. Respectively the false negatives and true negatives can be interpreted in the same way.

	Actually Positive	Actually Negative
Predicted Positive	True Positives (TPs)	False Positives (FPs)
Predicted Negative	False Negatives (FNs)	True Negatives (TNs)

Figure 34: Confusion Matrix

The reason to set this focus is that the user should be confident to perform an action once the algorithm detects stiction. For example, if a false positive result occurs and the user performs an action based on that detection. This will lead to a loss in resources and the user will lose the trust in the accuracy of the algorithm.

4.1 The Validation Data

In this section the data to test the algorithm will be presented. The industry use cases were taken from the South African Council for Automation and Control database (SACAC) (Bauer, et al., 2018). The database contains information about several control valves providing common valve issues. These issues are quantization, saturation, sensor fault, stiction, tuning problems and a section with unknown cases. The main part of the testing will be about how to detect the stiction cases. In the process industry it can occur that data is wrong labelled. For that reason, all of the seven cases labelled as stiction have to be confirmed. The controller output signal from all stiction cases were collected in Figure 35.

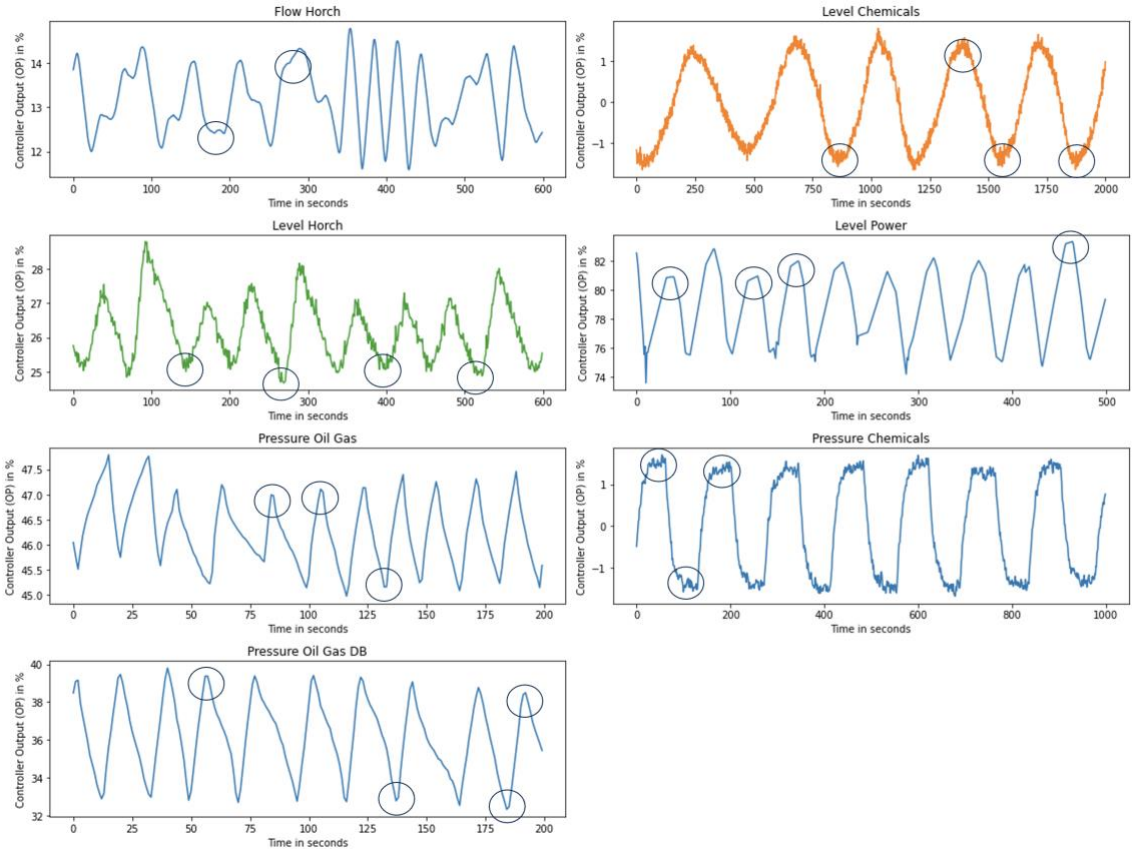


Figure 35: Stiction cases in the SACAC database

All the signals in Figure 35 are clearly oscillating and are therefore able to fulfill the first requirement. The next indication of stiction are the flat lines in the peaks of the signals. The explanation why these are clear signs of stiction can be found in Chapter 2.2.2. Therefore, all of the graphs were examined for flat areas. When it is detected, a blue line is drawn around it. All signals contain flat peaks and will be further investigated. The next step is to investigate if these flat areas lead to

sharp cornered ellipses. The Figure 36 shows the ellipses to the corresponding cycles. It would be too extensive to investigate all the ellipses. The focus is set on the first flat peak that was encountered.

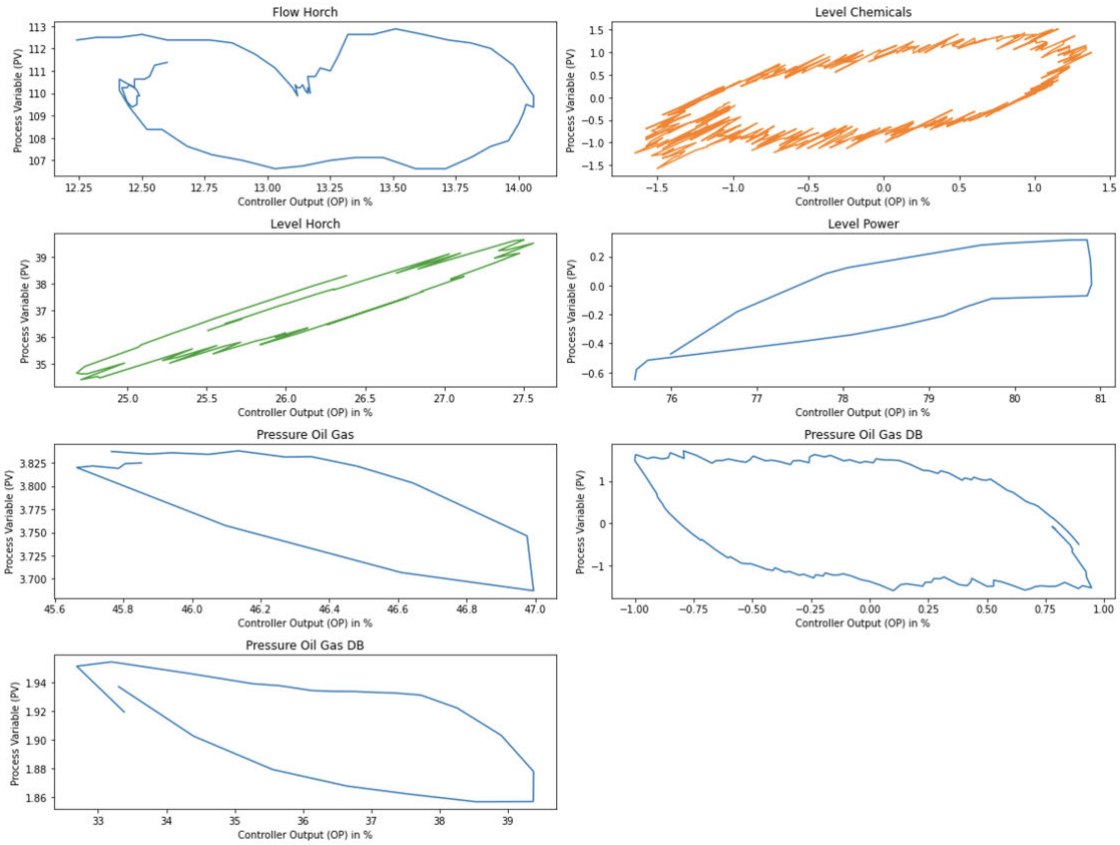


Figure 36: Second gate stiction ellipses

Six of the seven ellipses seem to fulfill the requirement of sharp cornered ellipses. The case at the top left corner does not show a perfect ellipse but does fulfill the other criteria. Therefore, all the ellipses will be used to validate the algorithm. Additionally, the different severeness's of noise can be observed. The ellipse at the top right corner shows that the signal suffers from strong noise. Whereas the ellipses at the bottom left corner have a relative low noise level. It would be difficult to detect an ellipse contained with noise even if there is one. This task seems easy for a human but is complex for a computer. The need of an automatic detection of the noise and the need to select a filtering parameter is once more proven.

4.2 Performance of the Algorithm

In this section the accuracy of the algorithm will be investigated. As presented above the validation data are taken from the SACAC database. The validation cases are examples of the industry that contains stiction issues in the control valve. At last, the non-stiction cases will be shown. To analyse all the examples in detail would be too extensive. Therefore, the results of the algorithm will be summarized in a table.

The presentation of the results will be shown in the Seeq software. For the first signal the error and the OP data were selected in the dropdown menu. For this setup no condition was selected.

The parameter selection and the results are shown in Figure 37. In the result section it can be determined that one oscillation was found. In addition, the start and end time of this oscillations is shown. In the next column the amount of detected cycles are displayed with the additional information on how many of these cycles were detected as stiction. This column is followed by the apparent amount of stiction in percent. The first example shows clear characteristics of stiction. This use case was presented in Figure 35. The results of the add-on can be reviewed in Figure 37. There is an oscillation detected with 24 cycles. This oscillation contains 4 cycles with stiction characteristics, present in the results section.

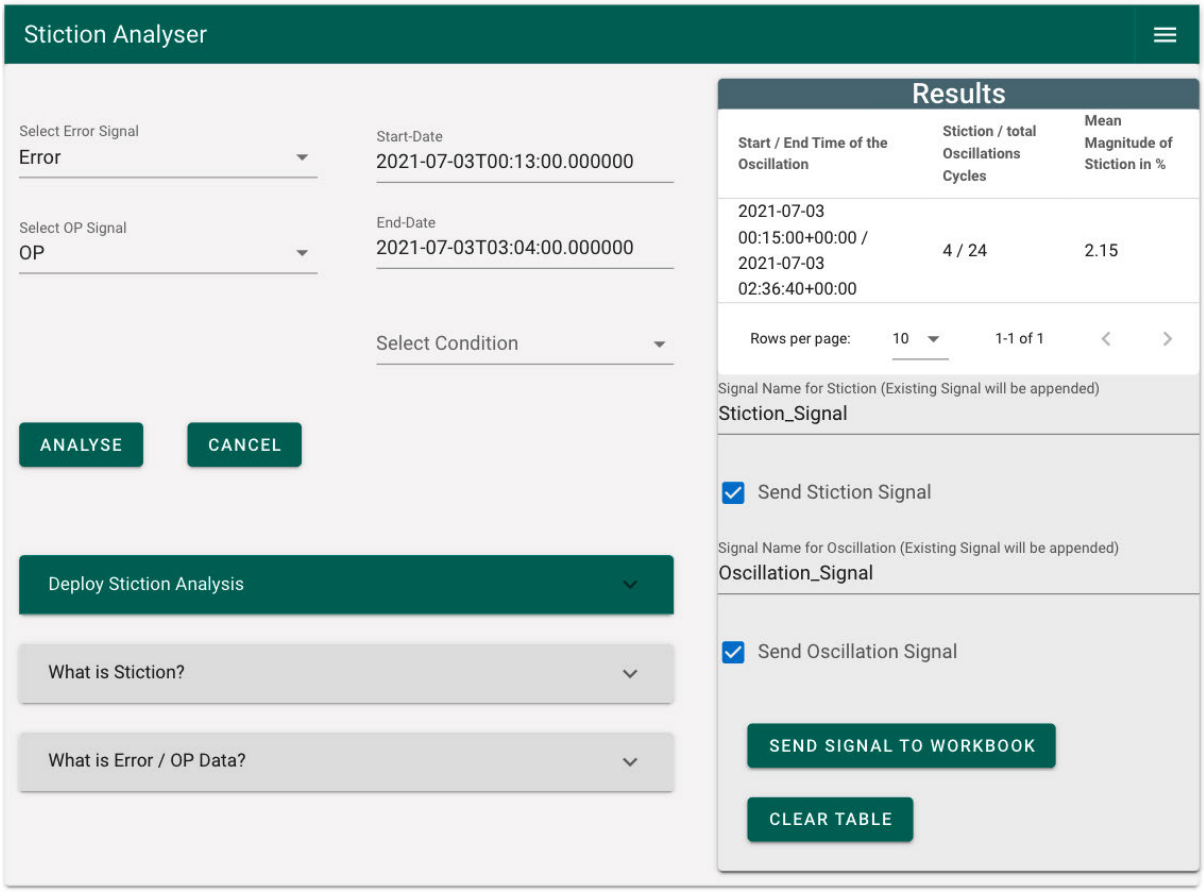


Figure 37: Selection for a stiction signal

The next step is to define a name for the oscillation and the stiction signal and to send it back to the Seeq Workbook. The results in the Workbook can be investigated in Figure 38. The purple signal is the error signal, the orange signal is the OP signal, the pink signal is the oscillation signal and the blue signal is the stiction signal. One can clearly see that the regular oscillating can be detected as well as the stiction characteristics. The apparent amount of stiction is with 2.15 % relatively low.

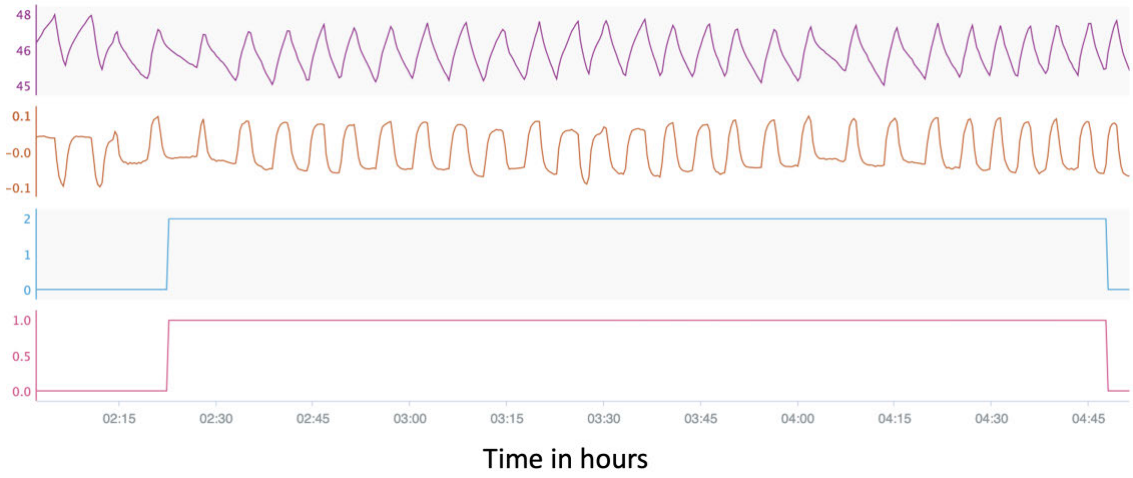


Figure 38: Stiction results

In order to validate the results from the algorithm the apparent amount of stiction will be investigated by hand. Thus, one cycle will be investigated which is illustrated in Figure 39.

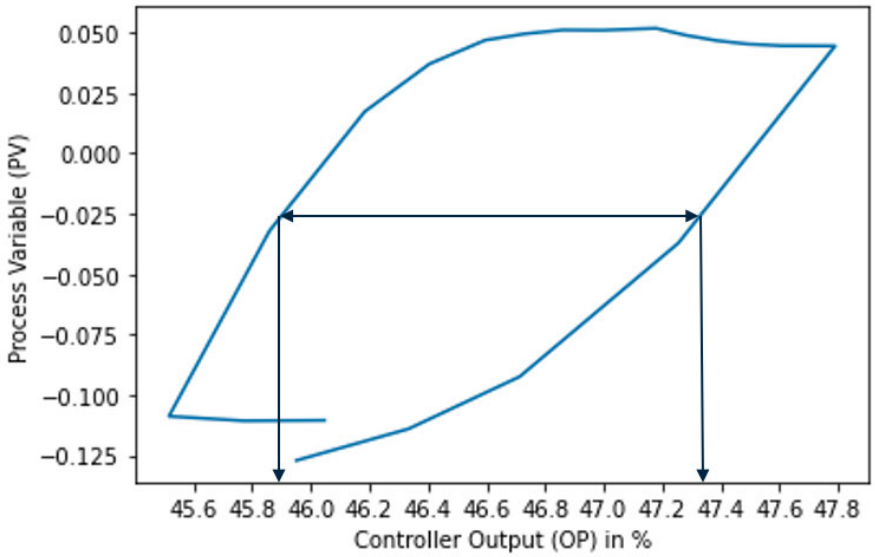


Figure 39: Width of the ellipse

The width of the ellipse was measured with 1.5 %. The value taken from the fitted ellipse is 2.15 %. Therefore, the accuracy for this case is about 70 %. The deviations can be explained because of the ellipse that has to fit into the shape.

The accuracy of the method is not reliable enough to be further discussed. In addition, the field examples has shown that an increase of the stiction amount within one signal cannot be measured. The reason for that is that the utilized functions to fit the ellipse delivers unreliable results. Therefore, a new algorithm

needs to be developed in order to achieve this objective. As the idea behind the approach is well suited to measure the apparent amount of stiction the fundamental concept will stay remain. Only the programming part shall be changed to increase the accuracy. The idea is to use the centre of the ellipse and to detect the left and the right point from the detected shape. These two points define the width of the ellipse. The distance can be calculated by subtracting the two values from each other. The results are the pixels between these two points. The problem is that the unit of the calculated values is not interpretable. This is shown in Figure 40.

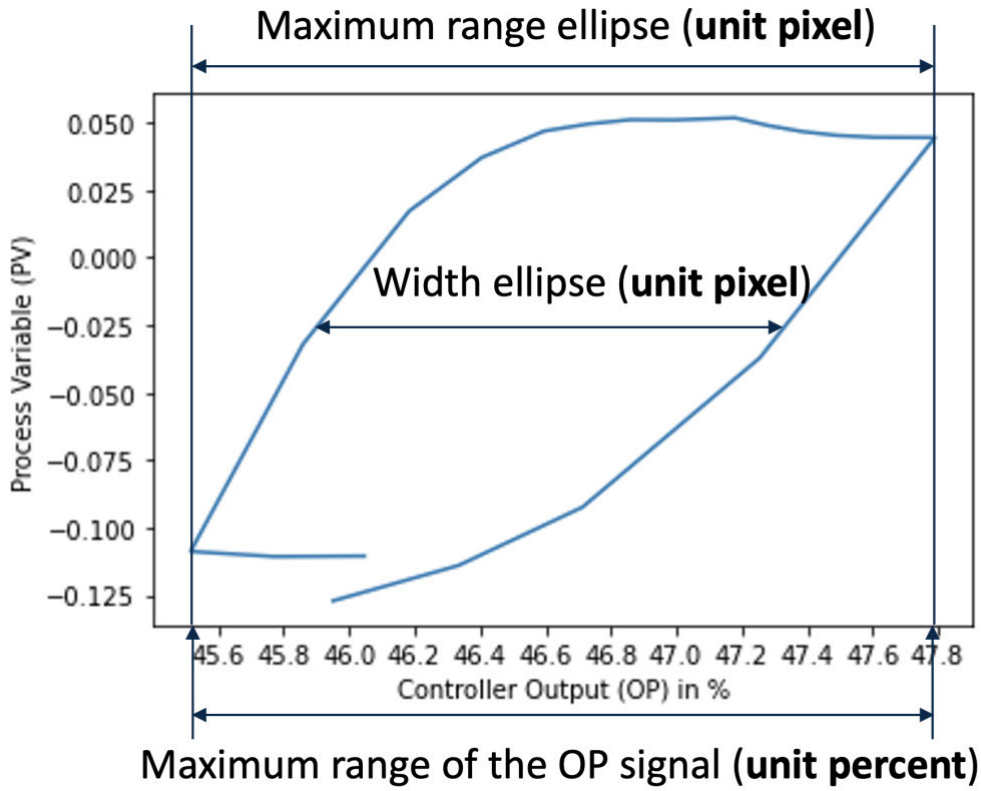


Figure 40: Unit problem in improved measurement algorithm

To achieve this target the minimum and maximum values in the x direction of the contour are determined. These two values are subtracted by each other to calculate the maximum distance. The previous calculated distance will then be divided by the maximum distance. The ration will end up in a value between zero and one. Now the minimum and the maximum of the OP signal, that was used to create the plot, are determined and subtracted by each other. The last step is to multiply the ratio with the range of the OP signal. The result will be the apparent amount of stiction in percent and is shown in the Table 2.

Table 2: Magnitude measurement – Comparison of the Results

Name	Mean magnitude in %	Actual magnitude in %	Measurement accuracy in %	Old measurement accuracy in %
Flow Horch	2.81	2.9	97	55
Level Chemicals	1.79	1.89	81	95
Level Horch	-	1	-	1
Level Power	1.68	2.7	62	76
Pressure Oil Gas	1.03	1.4	74	70
Pressure Chemicals	1.48	1.75	85	48
Pressure Oil Gas DB	3.71	4.7	79	57

In two cases the accuracy has decreased of about 14 %. Whereas in three cases it improved over 30 %. The improved algorithm delivers more accurate results and it is able to measure the width of any ellipse. Even in cases where the ellipse is oddly shaped the measurement stays more accurate than before. Furthermore, the algorithm is able to measure an increase in the stiction signal.

It is too extensive to present all the stiction cases from the SACAC database in detail. Thus, the results for the stiction cases are shown in the Table 3.

Table 3: Stiction detection results

Name	Stiction / total oscillation cycles	Stiction	Measurement accuracy in %
Flow Horch	1 / 16	Yes	97
Level Chemicals	1 / 38	Yes	81
Level Horch	0 / 9	No	-
Level Power	2 / 169	Yes	62
Pressure Oil Gas	4 / 24	Yes	74
Pressure Chemicals	2 / 4	Yes	85
Pressure Oil Gas DB	2 / 12	Yes	79

It can be determined that the algorithm was able to detect six of the seven stiction cases sufficiently. The case that was not detected has to be further investigated. The time trend of the signals and the error OP plot is demonstrated in the Chapter

4.1. The first gate of the analysis is to detect an oscillation. This task was sufficiently achieved by the algorithm. Further, the edges of the ellipse needs be analysed regarding to their sharp corners. It seems that the algorithm was not able to detect a sharp corner. The reason could be that the width of the ellipse is too small and the algorithm detects this area as a single line instead of a triangle.

The total accuracy to detected stiction is therefore about 86 %. For the measurement accuracy the results seemed to be reliable in most of the cases. This measurement was developed to order stiction cases on a wider perspective.

The next step is to test if the algorithm does detect other valve issues as stiction. The results can be investigated in the Table 4.

Table 4: Results of other valve issues

Name	Valve issue	Oscillating	Stiction / total oscillation cycles	Mean magnitude in %
Level Minerals	Saturation	No	-	-
Temperature Oil Gas	Saturation	Yes	0 / 16	-
Flow Oil Gas	Sensor fault	Yes	3 / 53	4.73
Flow Chemicals	Tuning	No	-	-
Level Horch	Tuning	No	-	-
Quality Horch	Tuning	Yes	0 / 32	-

Only one of the non-stiction labelled cases was detected as stiction. This case needs to be further analysed to understand the limitations of the algorithm. The Figure 41 shows the time trend of the signals presented in the Seeq software. By investigating the oscillating OP signal it can be derived that there are flat areas.

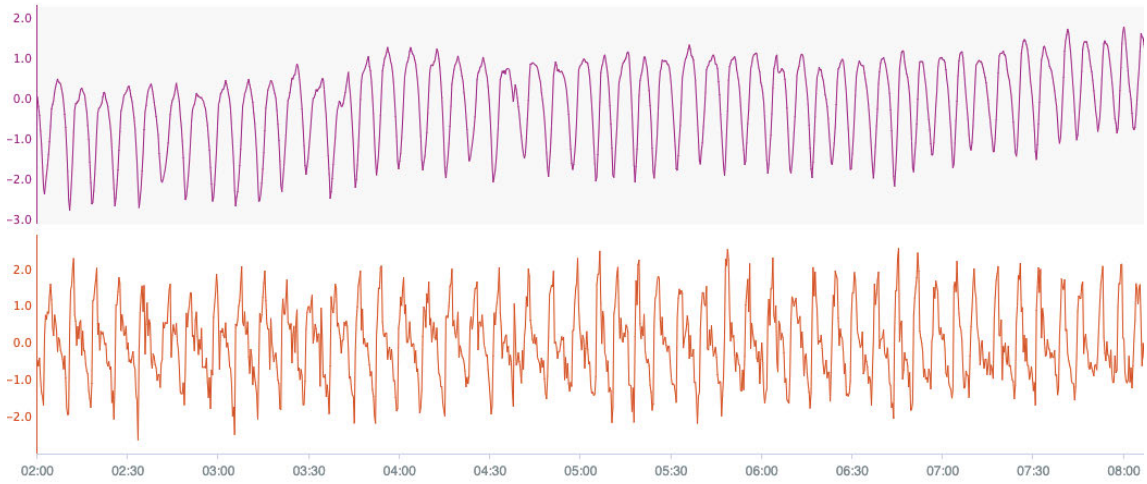


Figure 41: Error and OP signal from a non-stiction case (sensor fault)

Thus, a single cycle will be investigated to see if there are sharp cornered ellipses. The error OP plot is illustrated in Figure 42.

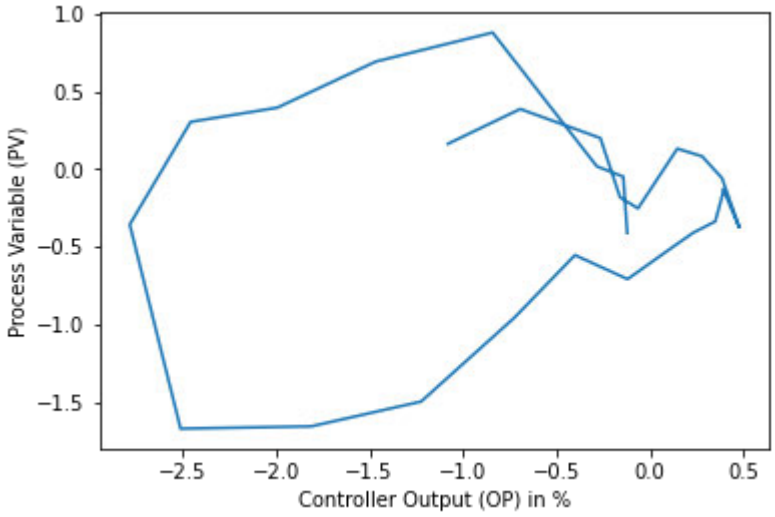


Figure 42: Error and OP plot from a non-stiction case (sensor fault)

The ellipse in the image is clearly a sharp cornered ellipse and therefore the algorithm detected what it should do. At this point it is not clear if several valve issues came together in this example, but one can derive that stiction is one of the issues. In fact, this case is not a sufficient representation of a non-stiction case and it will not be included in the accuracy calculation.

In conclusion one can derive that the stiction detection accuracy is about 86 %. Furthermore, none of the obvious non-stiction cases were detected as stiction. Thus, the requirement to reduce the false positive results is fulfilled. However, the total accuracy of the algorithm is about 92 %.

5 Conclusion and Future Work

The given task was to develop an algorithm that is able to detect shapes in scatter plots of industrial data. The requirements on the algorithm were that the user has a minimal amount of parameters to select. Thus, the algorithm has to operate automatically without the interaction of the user. Furthermore, a crucial point was to minimize the number of false positive results. A high false positive rate in the results of the algorithm would decrease the trust of the user. Leading to a situation where the user would not use the algorithm further.

The initial step for the development was to find characteristic signs for valve stiction in scatterplots. Currently none of the stiction detection approaches use a lean approach for pattern recognition in images. Therefore, a new algorithm has to be developed within this work.

First it was determined that a stiction case always led to an oscillating behaviour in the process data. Thus, an oscillation detection algorithm was used to detect the start and end point for this oscillation. The oscillation will be detected if the signal oscillates around the setpoint. The fundamental idea is to use the zero crossing to detect if a predefined number of cycles could be reached. However, the next step is to find characteristic signs of stiction in scatterplots. For the characteristic signs the error OP plot seemed to be sufficient. In the error OP plots an oscillation ends up in an ellipse. Whereas a signal that is suffering from stiction ends up in a sharp corner of the ellipse. The challenge was to develop a workflow that is able to differentiate between a normal and a sharp corner ellipse without the use of a highly complex algorithm.

In order to fulfill this task several algorithms from the field of computer vision were taken. These algorithms were combined to create a unique workflow which is able to distinguish between a sharp corner and a round corner ellipse. In the background, the algorithm detects the shape in the image and compare it with ellipses from a database. A threshold was set to define the shapes that are similar enough to pass the gate. Next, the edges will be cropped and another algorithm tries to reduce the number of necessary points to represent the shape. Three points are a clear indication that the shape in the image is a triangle. Thus, the ellipses have sharp corners.

The algorithm was tested with the data from the SACAC database. This database contains several examples of valve issues from different industries. In order to test the robustness of the algorithm all the available examples were taken for testing purposes. It was determined that the algorithm could detect six of the seven stiction examples leading to an accuracy of about 86 %. The magnitude of the one undetected case was one percent %. In fact, this was not a severe case of stiction.

Further, none of the clear non-stiction cases were detected as stiction. The total accuracy of the algorithm is therefore about 92 %. The width of the detected ellipse will be calculated for the measurement of the magnitude of the stiction. The width gives the apparent amount of stiction as the error and the OP data contain the process dynamics. The results were sufficient for any shape with a similarity to an ellipse.

All in all, a three-gate system was developed to fulfill the requirement. Starting with the oscillation detection, followed by the shape detection and the similarity verification. At last, the reduction of the needed points is proceeded. Followed by the triangle detection. The total accuracy is outstanding under consideration that even different level of noisy data can be detected. All of these results were achieved without any parameter input form the user. Only the analysed signals have to be set in.

5.1 Future Work

The developed algorithm is a clear proof that it is possible to use computer vision algorithm in the process industry for process monitoring. The scope of the algorithm was to detect stiction in control valves. The next goal to reach would be to develop a more general shape detector. The user should be able to define a certain number of shapes which will then be searched in the provided data. The shapes could represent crucial process conditions. The algorithm would afterwards search for the labelled states and would help the operator as well as the process engineers to run the plant. One idea is to develop an SQL database to store the inserted shapes. The inserted shapes would now be available for the whole team or even the entire organisation. This step would help to share important findings and it lays a foundation of knowledge. Furthermore, the shape

detector would help to analyse the high amount of process data in a smaller amount of time. As time is of importance in the process industry, this can lead to a competitive advantage.

6 Bibliography

Ang, K., Chong, G. & Li, Y., 2005. PID control system analysis, design, and technology.. IEEE Transactions on Control Systems Technology, pp. 559 - 576.

Jelali, M., 2007. Estimation of valve stiction in control loops using separable least-squares and global search algorithms. ScienceDirect, Volume 18, pp. 632-642.

Suzuki, S. & Abe, K., 1985. Topological Structural Analysis of Digitized Binary Images by Border Following. Academic Press, Inc, Volume 30, pp. 32-46.

Shin, W. & Marquez, M. R. G., 2003. A non-self-intersection Douglas-Peucker Algorithm. IEEE.

Douglas, D. H. & Peucker, T. K., 1973. ALGORITHMS FOR THE REDUCTION OF THE NUMBER OF POINTS REQUIRED TO REPRESENT A DIGITIZED LINE OR ITS CARICATURE. THE CANADIAN CARTOGRAPHER, Volume 10, pp. 112 - 122.

Hu, M.-K., 1961. Visual Pattern Recognition by Moment Invariants. PGIT.

Choudhury, M. S., Jain, M. & Shah, S. L., 2007. Stiction – definition, modelling, detection and quantification. ScienceDirect, Volume 18, pp. 232 - 243.

Hägglund, T., 1995. A Control-Loop Performance Monitor. Elsevier Science Ltd, Volume 3, pp. 1543 - 1551.

Thornhill, N. F., 2005. Finding the Source of Nonlinearity in a Process With Plant-Wide Oscillation. IEE, Volume 13, pp. 434 - 443.

Thornhill, N. F. & Hägglund, T., 1997. Detection and Diagnosis of Oscillation in Control Loops. Elsevier Science Ltd, Volume 5, pp. 1343 - 1354.

Choudhury, M. A. A. S., Thornhill, N. F. & Shah, S. L., 2004. A Data-Driven Model for Valve Stiction. ADCHEM.

Zabiri, B. K. H. et al., 2020. A simple model-free butterfly shape-based detection (BSD) method integrated with deep learning CNN for valve stiction detection and quantification. ELSEVIER Journal of Process Control, Volume 87, pp. 1 - 16.

Amiruddin, A. A. A. M. et al., 2019. Valve stiction detection through improved pattern recognition using neural networks. ELSEVIER Control Engineering Practice, Volume 90, pp. 63 - 84.

Vazquez, N. R., Fernandes,, D. P. & Chen, D. H., 2019. Control Valve Stiction: Experimentation, Modeling, Model Validation and Detection with Convolution Neural Network. International Journal of Chemical Engineering and Applications, Volume 10.

Henry, Y., Aldrich, C. & Zabiri, H., 2020. Detection and severity identification of control valve stiction in industrial loops using integrated partially retrained CNN-

PCA frameworks. ELSEVIER Chemometrics and Intelligent Laboratory Systems, Volume 206.

Bacci di Capaci, R. & Scali, C., 2018. Review and comparison of techniques of analysisof valve stiction: From modeling to smart diagnosis. ELSEVIER Chemical Engineering Research and Design, Volume 130, pp. 230 - 265.

Taqvi, S. A. A. et al., 2020. A Review on Data-Driven Learning Approaches for Fault Detection and Diagnosis in Chemical Processes. Wiley .

Brásio, A. S. R., Romanenko, A. & Fernandes, N. C. P., 2014. Modeling, Detection and Quantification, and Compensation of Stiction in Control Loops: The State of the Art. ACS Puplications, Volume 53, pp. 15020 - 15040.

Choudhury, M. S., Thornhill, N. & Shah, S., 2005. Modelling valve stiction. ELSEVIER Control Engineering Practice, Volume 13, pp. 641 - 658.

Kano, M., Maruta, H., Kugemoto, H. & Shimizu, K., 2004. PRACTICAL MODEL AND DETECTION ALGORITHM FOR VALVE STICTION. IFAC.

Kvam, A., 2009. Detection of Stiction in Control Valves an Algorithm for the Offshore Oil and Gas Industry. Norwegian University of Science and Technology.

Fisher, 2005. Control Valve Handbook. s.l.:Emerson Process Managment.

Hägglund, T., 1995. A CONTROL-LOOP PERFORMANCE MONITOR. Elsevier Science Ltd , Volume 3, pp. 1543-1551.

Zheng, D. et al., 2021. Valve Stiction Detection and Quantification Using a K-Means Clustering Based Moving Window Approach. ACS Publications, Volume 60, pp. 2563 - 2577.

Wang, Z. & Bovik, A. C., 2002. A universal image quality index. IEEE Signal Processing Letters, Volume 9, pp. 81 - 84.

Wang, Z., Bovik, A. C., Sheikh, H. R. & Simoncelli,, E. P., 2004. Image Quality Assessment: From Error Visibility to Structural Similarity. IEEE TRANSACTIONS ON IMAGE PROCESSING, Volume 13, pp. 600 - 612.

Butterworth, S., 1930. On the Theory of Filter Amplifiers. The Wireless Engineer, pp. 536 - 541.

Hägglund, T., 2011. A shape-analysis approach for diagnosis of stiction in control valves. Elsevier Control Engineering Practice , Volume 19, pp. 782 - 789.

Bauer, M. et al., 2016. The current state of control loop performance monitoring - A survey of application in industry. J. Process Control, Volume 38, pp. 1 -11.

Bauer, M., Auret, L., Le Roux, D. & Aharonson, V., 2018. ,An industrial PID data repository for control loop performance monitoring (CPM). IFAC-PapersOnLine, Volume 51, pp. 823-828.

Hu, J., Cai, S. & Zhang, L., 2017. Alarm Management Technology and Its Progress in Process Industries. Atlantis Press, Volume 134.

7 Appendix

In the following section additional documents are listed.

7.1 Installation Guideline

seeq-stictionanalyser is a Python module to detect oscillations and stiction patterns in control valves. It is intended to be used as an add-on in the Seeq Workbench. The oscillation analyser, which is part of this add-on, detects and isolates periods of time during which an oscillation occurs. The stiction analyser identifies if stiction is present. Those stiction cases have to be within the oscillating time periods identified by the oscillation detector. An oscillation index and/or a stiction index can be pushed back into the Seeq Workbench as time series signals for monitoring and analysing (e.g. with the "Value Search" function in Seeq). The module includes a user interface (UI) designed to interact with the Seeq server.

7.2 User Guide in the Installation

seeq-stictionanalyser User Guide provides a more in-depth explanation of the algorithm behind the stiction analysis and how the seeq-stictiondetection works. Examples of typical types of analyses using **seeq-stictionanalyser** can be found in the User Guide.

7.2.1 Installation

seeq-stictionanalyser requires **Python 3.7** or later.

7.2.2 Dependencies

See requirements.txt file for a list of dependencies and versions. Additionally, you will need to install the Seeq module with the appropriate version that matches your Seeq server. For more information on the Seeq module see Seeq at pypi.

7.2.3 User Installation Requirements (Seeq Data Lab)

If you want to install **seeq-stictionanalyser** as a Seeq add-on Tool, you will need:

- Seeq Data Lab (>= R50.5.0, >=R51.1.0, or >=R52.1.0)
- Seeq module whose version matches the Seeq server version
- Seeq administrator access
- Enable add-on Tools on the Seeq server

7.2.4 User Installation (Seeq Data Lab)

The latest stand of the project can be found in pypi.org as a wheel file. The file is published as a courtesy to the user and it does not imply any obligation for support from the publisher.

1. Create a new Seeq Data Lab project and open the terminal window
2. Run pip install stictionanalyser
3. Run python -m seeq.addons.stictionanalyser [--users <users_list> --groups <groups_list>]

7.2.5 Development

We welcome new contributors of all experience levels. The **Development Guide** has detailed information about the workflow, documentation, tests, etc.

7.2.6 Important Links

- Official source code repo: https://github.com/HAW-Process-Automation/Stiction-Analyser/
- Issue tracker: https://github.com/HAW-Process-Automation/Stiction-Analyser/issues

7.2.7 Source Code

You can get started by cloning the repository with the command:

git clone git@github.com:HAW-Process-Automation/Stiction-Analyser.git

7.2.8 Installation from the Source

For the development work, it is highly recommended to create a python virtual environment and to install the package in that working environment. If you are not familiar with python virtual environments, you can take a look here.

Once your virtual environment is activated, you can install **seeq-stictionanalyser** from source with:

python setup.py install

7.2.9 Testing of the Package

There are several types of testing available for the **seeq-stictionanalyser.**

7.2.10 Automatic Testing

After the installation, you can launch the test suite from the root directory of the project (i.e. seeq-stictionanalyser directory). You will need to have pytest >= 5.0.1 installed.

To run all tests:

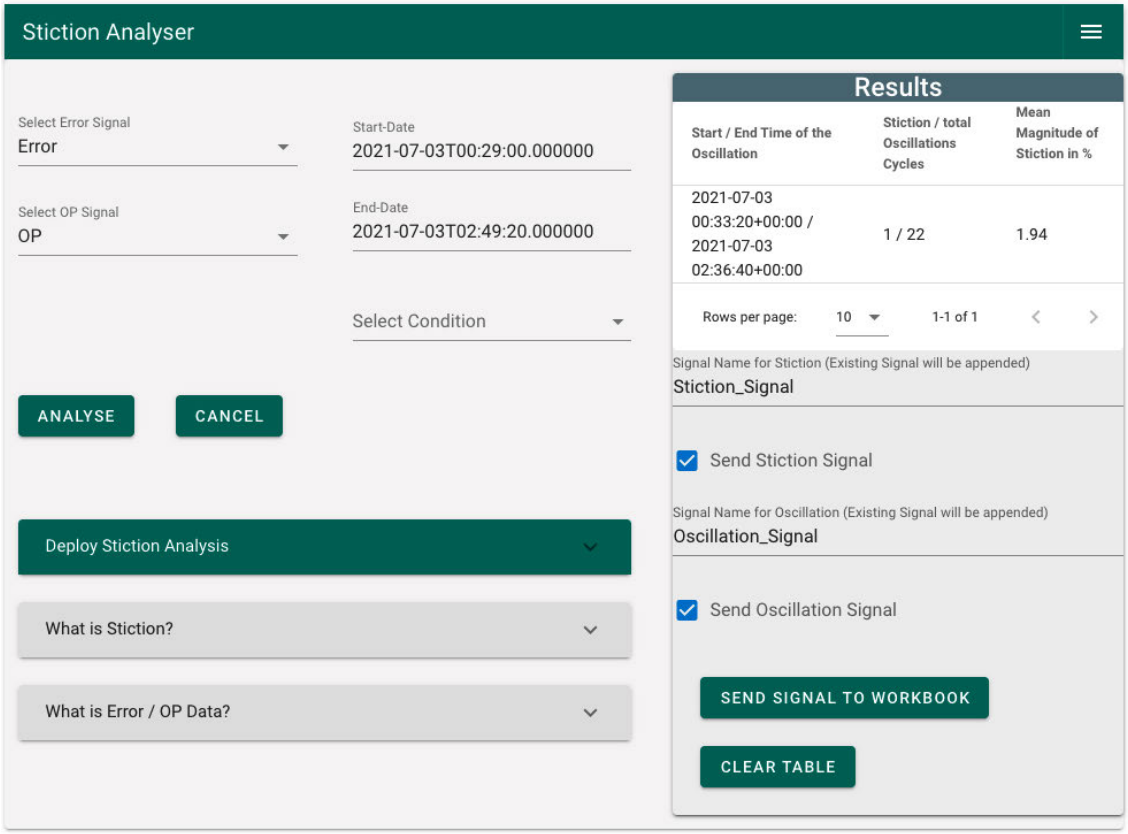
pytest

Note: Remember that the Seeq module version in your local environment should match the Seeq server version.

7.2.11 Test of the User Interface

To test the UI, use the developer_notebook.ipynb in the development folder of the project. This notebook can also be used while debugging from your IDE. You can also create a .whl first, install it on your virtual environment and then run developer_notebook.ipynb notebook there.

7.3 User Guide



7.3.1 Overview

This section explains the workflow and the algorithm of the seeq-stictiondetection add-on. It also includes important technical terms and the physical background will be explained.

7.3.2 What is Valve Stiction and why should it be detected?

Every plant in the process industry requires the ability to control important variables such as temperatures and flows during production to ensure a safe manufacturing process. Actuators are field instruments that act on the process. Valves are a type of an actuator that can increase or decrease the amount of liquids and gases, for example to cool a reactor. Valves directly intervene in the process. With the nearly infinite possible use cases in the process industry valves are one of the most common control apparatus. A good operation of the valves is therefore of utmost importance. To ensure an unobstructed manufacturing process it is of importance to monitor the performance of the valves. One possible way is to investigate the data received from the process monitoring system and try to find characteristic patterns inside the data. Here, we first explain the basics of automation and control loops to understand which data is available and how it can help to monitor the control valves.

7.3.3 Control Theory

In this section the fundamentals of control will be explaining with a simple feedback control loop. For instance, the control loop could handle a certain temperature in the process. The temperature will be measured with a sensor and the valve is the actuator. The process variable (PV) is compared to the desired setpoint (SP). The controller operates on the difference of the signals (error) between the PV and the SP. The signal (OP) is given to the actuator, which is in this case the valve. The valve output is a flow which acts on the process. In theory, the valve changes its values according to the OP signals. In reality several problems can occur. It is important to note that in the process industry the three variables SP, PV and OP are recorded for each control loop. The single input

single output control loop is illustrated in Figure 1. The manipulated variable (MV) is usually not recorded and therefore not accessible for an analysis.

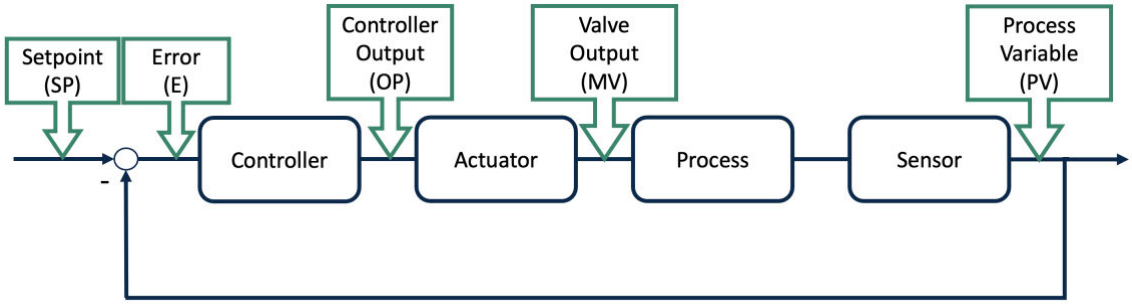


Figure 1: Feedback loop

An indication of a problem with the valve is valve stiction. Stiction causes an oscillation in the process variable (PV) and the controller output (OP). The reason is that the valve stem does not directly transform the controller output to the manipulated variable. To distinguish between oscillations caused by stiction and oscillations caused by other problems in the control loop, such as tight tuning settings, one need to look at the shape of the oscillation. Pure sinusoidal oscillations will result in a perfect shape of the ellipse when plotting the PV against the OP. Oscillations caused by valve stiction will have sharp corners in the PV-OP plot. A setpoint change (SP) can change the shape of the PV-OP plot. Thus, it is advisable to use the error signal instead of the PV. The error is computed from subtracting the PV from the SP. In short, oscillations are identified and the error-OP plot is scanned for sharp corners.

7.3.4 Stiction Detection Algorithm

As explained in the section above, the detection algorithm should first detect oscillations and in case the oscillations are sharp cornered in the error OP plot, the valve shows stiction with a high probability. The algorithm for the oscillation detection examines and counts the zero crossings. Additionally, the approach evaluates if the amplitude of one cycle (data between two zero crossings) is above a user defined percentage. If the cycle can fulfill this requirement the cycle will be counted. If the number of counted cycles exceeds the predefined number within the supervision time then this time period contains an oscillating signal.

The algorithm was taken to determine the start and end point of the oscillation and to slice the raw signal. Then the sliced signal will be further sliced. The new sliced signal contains one full oscillation cycle. The next step is to plot the data in a scatterplot. Then an image will be created out of the plot. The following figure can be taken as an example on how the images could look like.

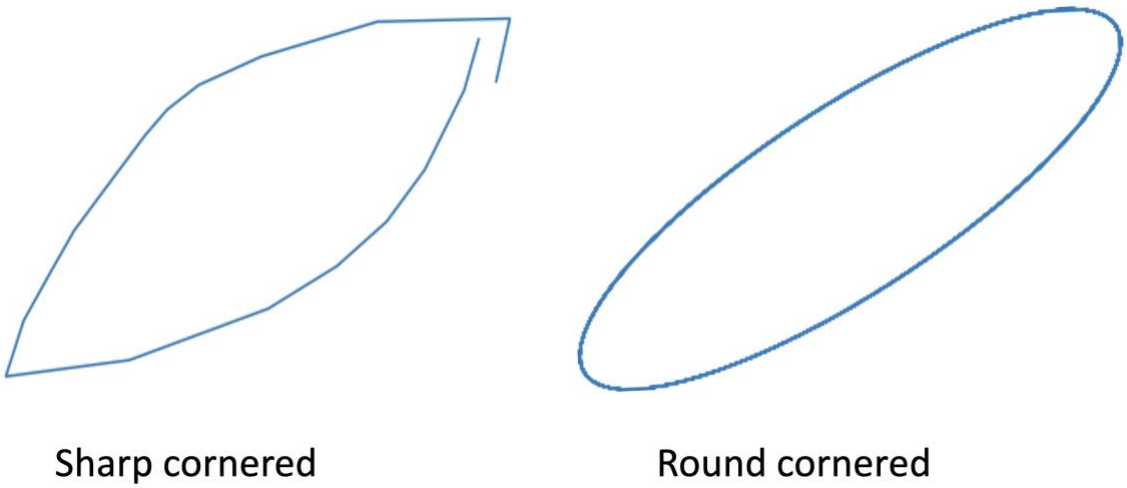


Figure 2: Sharp cornered ellipse and round cornered ellipse

A shape detection algorithm is applied to ensure that the investigated images contain an elliptical shape. For that the OpenCV library is utilized by first finding the contours and then comparing the ellipse with three different ellipses. In case the similarity with one of the compared ellipses is higher than 50 % the analysis will be continued. In the next step the two edges of the ellipse will be cropped and stored as separated images. This step is followed by closing and detecting the contour of the shapes. The images could now look like the ones illustrated in the following figure.

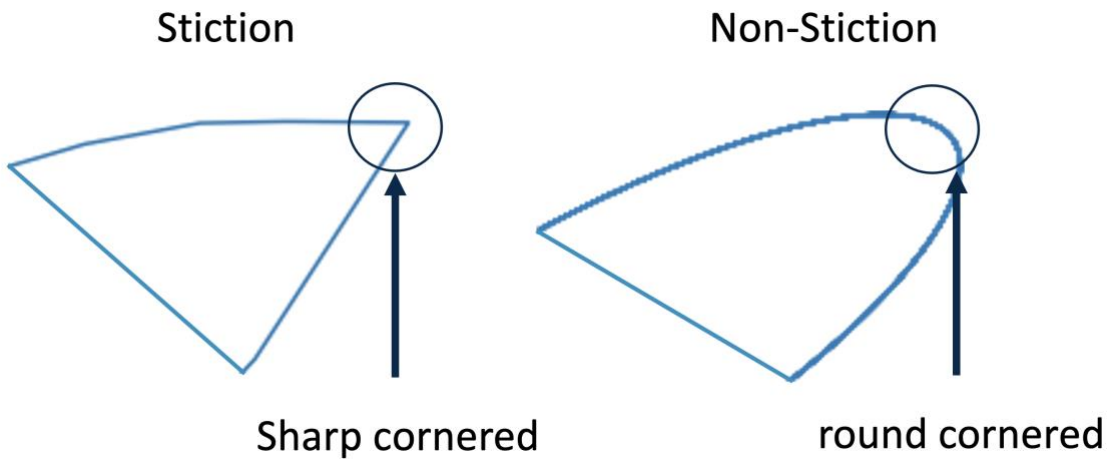


Figure 3: Sharp cornered ellipse compared to round cornered ellipse detail

To find the minimum number of points the Ramer–Douglas–Peucker Algorithm is applied. By connecting these points with straight lines, the shape will represent the original shape sufficiently. If the shape could be presented by three points it shows a triangle which means that it is a sharp cornered ellipse and therefore and clear evidence of stiction in the valve. The result of the analysis is then a signal which contains the apparent amount of stiction in percent. In addition, the user could send the oscillating signal back to the Workbench. The oscillating signal equals 1 if an oscillation is present and 0 if no oscillation is detected. The user could now apply a “Value Search” to find the time intervals where a signal is oscillating or is above a certain amount of stiction. The magnitude of the stiction will be calculated by fitting an ellipse in the sharp cornered ellipse. Then the width of the fitted ellipse will be calculated which represents the apparent amount of stiction in the control valve.

7.3.5 How to Use

The workflow can be divided into eight steps. These steps will be explained in detail within the following section. The workflow in short is given below.

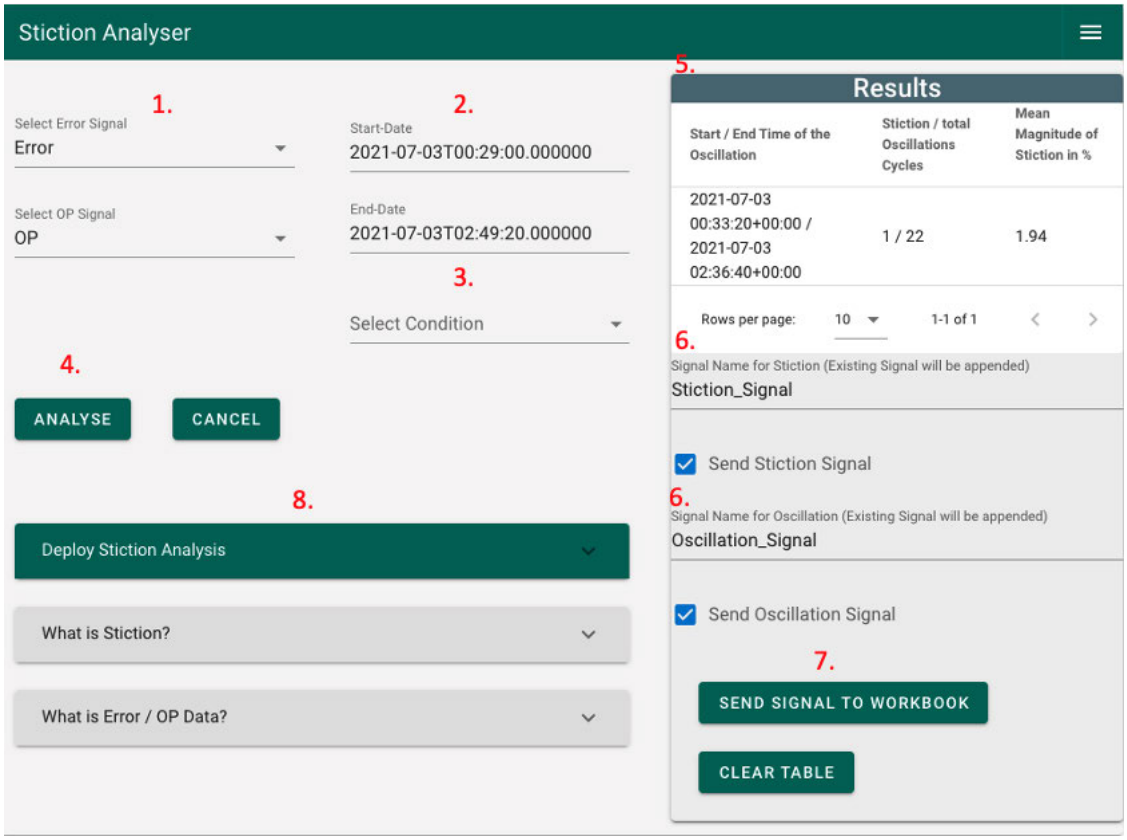


Figure 4: Stiction Analyser UI

7.3.6 Workflow

1) Select error and OP signal
2) Review if the selected time range is sufficient (the default is the display range)
3) Select a condition if needed (a selection of the wrong condition can be changed by clicking on the “CANCEL” button)
4) Click on the “ANALYSE” button (wait until the loader in the button disappears)
5) In case something interesting was found, it will be displayed in the “Results” section
6) Name the signals and check the boxes, which should be sent to the Workbench
7) Click on “SEND SIGNAL TO WORKBOOK” button
8) Use “Deploy Stiction Analysis” to schedule the stiction analysis (coming soon)

7.3.7 Example Use of Cases

7.3.7.1 First Use Case: Stiction Contained Signal (Level)

The first example containing information of stiction is taken from the SACAC Database and contains information about a level loop. A detailed look at the signal shows clear signs of stiction. These signs are illustrated in the following figure. A first indicator is that the signal is oscillating. The second indicator are the flat corners of the oscillation. These flat peaks are the reason for the sharp cornered ellipses in the error OP plot.

Figure 7: Detailed time trend of the level signal

The results of the algorithm are shown in the following figure. In this figure the error signal is orange, the OP signal is purple, the stiction signal is blue and the results of the oscillation finder is green. From the analysis one can see that the algorithm detected a stiction case with a magnitude of 2 %.

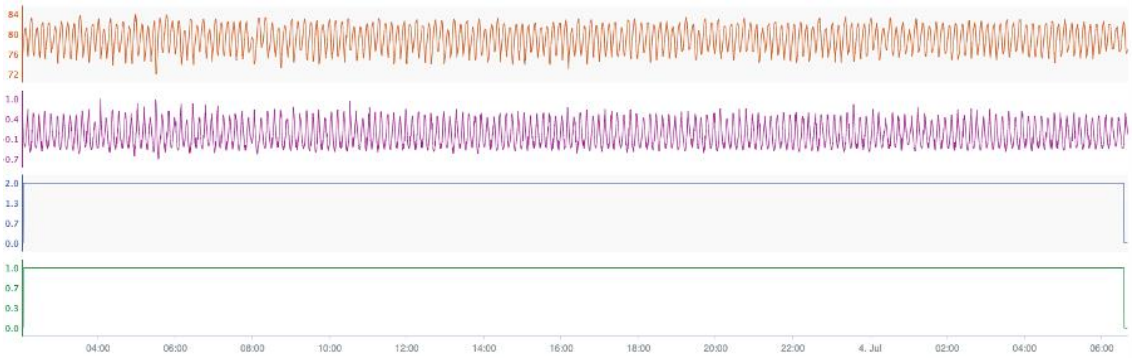


Figure 8: Results of the stiction analysis

7.3.7.2 Second Use Case: Stiction Contained Signal (Flow)

The third use case is a flow loop suffering from stiction. A slice of the signal can be investigated in the following figure. The flat corners of the OP signal are clear signs of stiction.

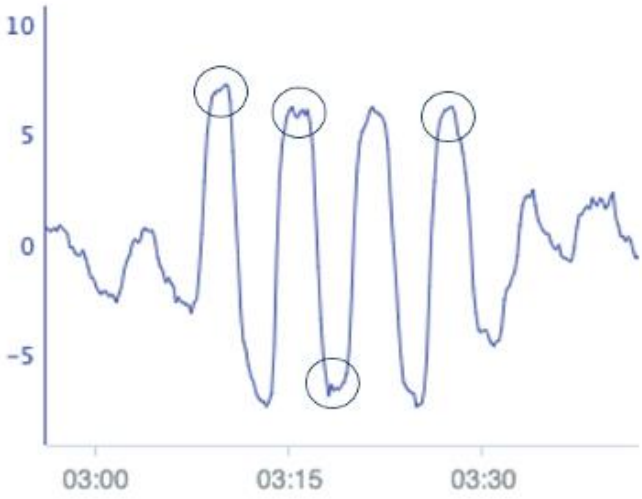


Figure 9: Detailed time trend of the flow signal

The results generated by the stictionanalyser add-on can be reviewed in the following figure. The add-on can detect the oscillation and the signs of stiction in the signals. In this figure the error signal is purple, the OP signal is blue, the stiction signal is orange and the results of the oscillation finder are green. The calculated apparent magnitude of stiction is 5 %.

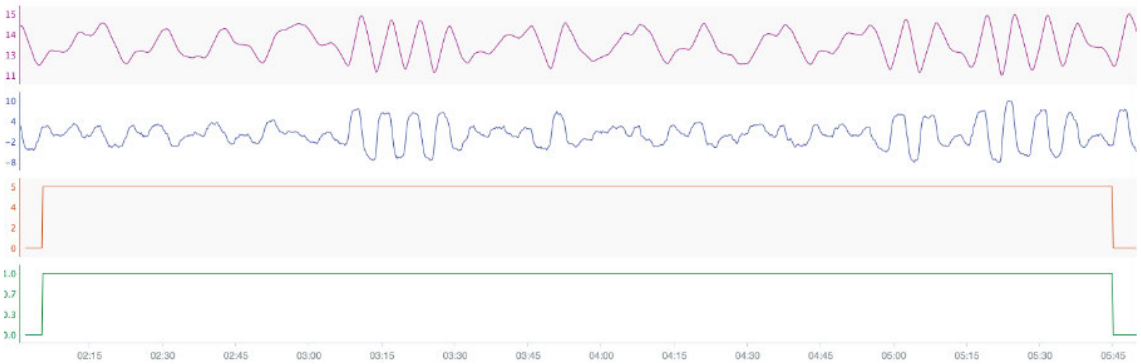


Figure 10: Results flow loop