Linköping University | Department of Management and Engineering
Bachelors's thesis, 18 credits| Mechanical Engineering
Spring 2024| LIU-IEI-TEK-G–24/02409—SE

Methods for Estimating the Pretension in Screw Joints

- With a focus on practical implementation and further development of an experimental impact drive rig

Emil Wallbom
Isak Röschmann

Supervisor: Martin Hochwallner
Examiner: Jonas Detterfelt

LINKÖPING UNIVERSITY

Linköpings universitet
SE-581 83 Linköping
+46 13 28 10 00 , www.liu.se

Abstract

Screw joints are one of the most prevalent construction elements. It is important to achieve the correct preload while tightening and to be able to control the preload present in an already tightened screw joint for the safety of the construction. An experimental impact drive rig built by previous bachelor projects is at the centre of this report. The rig was further developed, implementing new functions, and different behaviours of the rig were investigated. Different methods for estimating the preload in screw joints were investigated and realised, with a focus on using them with the experimental rig. Methods proposed were mostly based on angle, torque, and stretch measurements but also included ways of estimating the preload from the dynamic aspect of the impact rig. Not all of the proposed methods could be realised and analysed due to time restraints. One of the more promising methods was using bolt stretch to estimate the preload, utilising a specially designed screw length measurement rig designed by another group. The estimations using this rig would be suitable as a reference for other estimation methods. Further work should be directed towards making more measurements to evaluate the accuracy and spread of the different methods. The report also covers multiple valuable aspects for the continued work on the experimental rig.

Aknowledgements

This project would not have been possible without our supervisor and project owner, Martin Hochwallner. You have challenged us and pushed our understanding far beyond what we thought possible. We appreciated your interest in the project and the many valuable insights you have given us regarding the subject at hand. We also want to thank Nils Dressler for insightful conversations and access to special equipment we would not have been able to use otherwise. We are also thankful for getting to spend these months together with the two other groups. We have all helped each other to get through the project, and together we have learnt a lot.

The individual contribution to the project and the written report

Activity / Person	Emil Wallbom	Isak Röschmann
Plc Programming	100%	0%
Nutrunner	0%	100%
Working With the Impulse Drive	60%	40%
Investigating Estimation Methods	40%	60%
Realising Estimation Methods	50%	50%
Analysing Results	50%	50%
Written Part of the Report		
Introduction	50%	50%
Theory	40%	60%
Method	60%	40%
Rig Development	100%	0%
Rig Analysis	50%	50%
Realisation	50%	50%
Conclusion	50%	50%

Contents

List of Figures

List of Tables

Nomenclature

Symbol	Unit	Description
A	m^2	Area
$A_{threaded}$	m^2	Threaded Screw Area
β	°	Thread Half-Angle
d_1	m	Inner Thread Diameter
d_2	m	Mean Thread Diameter
E	Pa	Young's modulus
$E_{rotational}$	J	Rotational Energy
F_P	N	Preload
G	Pa	Shear Modulus
I	$kg\,m^2$	Rotational inertia
J	m^4	Polar moment of inertia
K	N/m	Stiffness
K_B	N/m	Bolt Stiffness
K_J	N/m	Joint Stiffness
$K_{torsion}$	$N\,m/rad$	Torsional stiffness
L	m	Length
ΔL	m	Elongation
μ_n	-	Friction Constant, Under Screw Head
μ_t	-	Friction Constant, Screw Thread
P	m	Thread Pitch
r_n	m	Screw Head Mean Contact Radius
r_t	m	Thread Mean Contact Radius
T	$N\,m$	Torque
$T_{tightening}$	$N\,m$	Tightening Torque
$T_{untightening}$	$N\,m$	Untightening Torque
θ_R	°	Rotational Angle
θ_{twist}	rad	Twist Angle
$U_{torsion}$	J	Potential Torsional Spring Energy
ω	rad/s	Angular velocity

1 Introduction

1.1 Background

Bolts are one of the most common fasteners and are used in a majority of constructions. They are highly standardised, and the bolts used today do not look far from what they did decades ago Bickford (2008). Yet they are still prevalent in the most high-tech structures and are an important part of enabling products to be re-manufactured to make greater use of our planet's natural resources.

The bolted joint works by creating a clamp force between two or more members. The clamp force is created by applying torque to the threaded bolt, the rotation of which stretches it, producing preload. During tightening, the goal is to achieve a great enough preload to keep the members from coming apart, whilst avoiding the tension in the bolt or the clamp force in the members exceeding the material's yield strength. However, there are a multitude of variables that make it difficult to achieve the desired preload. This leads to engineers having to oversize bolts to get around these uncertainties, leading to unnecessary costs and extra weight in construction. It is therefore of great interest to make accurate estimations of the preload achieved while tightening the bolted joint.

After tightening, the initial preload in the bolted joint is affected by the environment it is exposed to. This often results in a reduction of the preload in the joint. Because of this, estimating the preload in joints post tightening is also relevant.

In previous bachelor theses at Linköpings Universitet, an experimental rig for an impact drive has been built to analyse the tightening and untightening of screw connections. The rig is capable of tightening bolts through a sequence of impacts, but it has not yet been developed far enough to untighten bolts.

1.2 Purpose

The purpose of this work is, together with two other groups, to continue building upon and improving the current impact drive rig. This report aims to analyse, categorise, and realise different methods of estimating the preload in bolted fasteners during tightening and post-tightening. The other two groups focus on improving the current rig and improving the simulation models of the rig, respectively.

1.2.1 Research questions

- RQ1: How can one estimate the preload in bolted fasteners?
- RQ2: What methods could be implemented on the current impact drive rig?
- RQ3: What accuracy can be achieved by the different estimation methods?

1.3 Objectives

- Categorise methods for estimating preload.
- Evaluate practicality of different methods.
- Realise practical methods.
- Evaluate effectiveness of the realised methods.

1.4 Delimitations

This section will cover the delimitations that were set.

- When analysing the screw joint it is assumed that all deformations are elastic and no plastic deformation is taking place unless otherwise mentioned.
- Screw joints with one screw are the only ones that are investigated in this report.
- The phenomenon of relaxation and embedment of a joint after tightening will not be taken into account.

1.5 Structure of the report

This report will cover many different areas and topics. This section serves as an explanation of how the report will be structured. The introduction chapter will be followed by Method, Equipment and Theory, where all the necessary prerequisites to understand the rest of the work will be laid out. This includes information about the physical tools and test setups, the software, and the underlying equations that were used in the project. After

this, a short Categorisation chapter that goes over some categories for preload estimation methods is included. The next chapter presents the different preload estimation methods available, with some derived equations from the theory as well as how they could be implemented using different tools. This chapter is followed by one called Rig Development, where technical work done to improve the impact rig's capabilities is showcased. In the subsequent chapter, Testing and Analysing Rig Behaviour, some of the observed phenomena on the rig are tested and analysed. Next is the Realisation chapter where all the tested estimation methods with accompanying results, analysis, and discussion for each one are presented. This chapter begins by outlining how the methods were realised in general followed by the specific tests using these implementations. Since there are a handful of different realisations, these are numbered and include results, analysis, and discussion directly under each one.

2 Method

This chapter describes how the project was carried out, as well as establishing what software tools were used and provides a short explanation of how they work and how they were implemented.

2.1 Agile methodology

There are many ways of conducting a project with multiple individuals involved, one way is using an agile methodology. The agile methodology is a way of breaking up a project into smaller phases, ”sprints”, and dynamically focusing the work on the most important topics at hand (Laoyan, 2024). The method relies on periodic meetings between the sprints, during these the work done previously is evaluated and planning for what work should be carried out during the coming sprint is made.

During this project an agile methodology was implemented by the three bachelor thesis groups, meeting with the supervisor and project owner every other week. During these meetings the groups would present their progress and what they were currently working on. This made sure that everyone was up to date about the progress of the project as a whole. The meetings would end with all groups making sure that they have a clear path forward towards the project goal.

2.2 Method of work

In this section the workflow of the project is described. A flowchart with an overview of the project is shown in Figure 1 below.

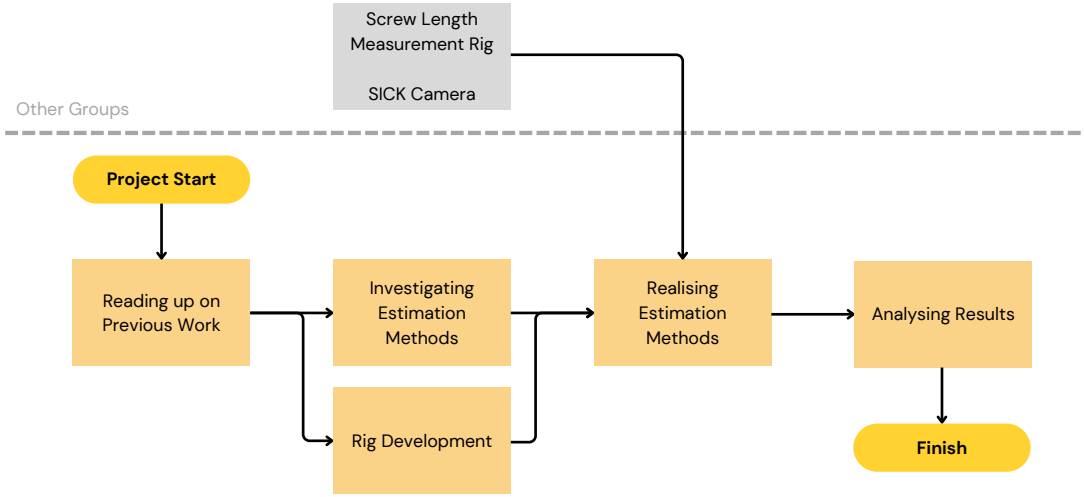

Figure 1: Flow chart visualising the different phases of the project.

Reading up on Previous Work There have been many bachelor projects covering the topic of screw tightening, which this project continues to build upon. This meant that there were already a lot of previous reports, existing equipment, and software that the group needed to get acquainted with.

Investigating Estimation Methods & Rig Development The work continued by investigating and categorising different methods of estimating the preload in a bolted joint. Planning of how these methods could be realised was also performed. In parallel with this, further development of the rig was carried out, improving its functionality.

Realising Methods When the rig was ready and the equipment designed by the Rig Improvement group was finished, the realisation of different methods began. All equipment was not ready at the start of this phase, which meant that different ”realisations” were made with different combinations of methods.

Analysing Results After the estimation methods had been realised, the results were then analysed.

3 Equipment and Software

In this chapter the equipment and software used in the project is presented.

3.1 Specimen

Figure 2: The specimen provided by previous bachelor thesis (Eriksson et al., 2022).

The specimen that was used in the project was developed by an earlier bachelor project. For more in-depth information and dimensions, see (Eriksson et al., 2022). The specimen in Figure 2 consists of a flange machined out of "Automatstål sexkant 36SMnPb14+C," which has been drilled and tapped to accommodate an M8 screw with a length of 100mm. The flange has a small hole drilled in the side of it, as the purpose of the previous bachelor project was to investigate the effect of corrosion on a screw joint.

In this project, two different specimens were used. They were reference specimens from the former project of Eriksson et al. (2022), and neither of the specimens had been subjected to corrosion. The specimen used to the greatest extent is Specimen A; this specimen has been tightened and untightened multiple times. The second specimen, Specimen B, was a fresh specimen that had only been tightened one time by the previous bachelor project.

The stiffness of the entire screw joint K_J and the screw itself K_B were calculated by Augustini et al. (2022) where FEM software was used to find the stiffness of the M8 bolt and the flange of the specimen. The result was then validated using equations from Verein Deutscher Ingenieure (2003). The stiffness values they achieved using the FEM software was used in this project. The stiffness for the 100 mm M8 bolt is $K_{\mathrm{B}} = 114.6\ \mathrm{MN/m}$ and for the entire screw joint $K_{\mathrm{J}} = 92.3\ \mathrm{MN/m}$. The screw's torsional stiffness K_{torsion} was also calculated by Augustini et al. (2022) to be 341 $\mathrm{Nm/rad}$.

3.2 Atlas Copco Nutrunner

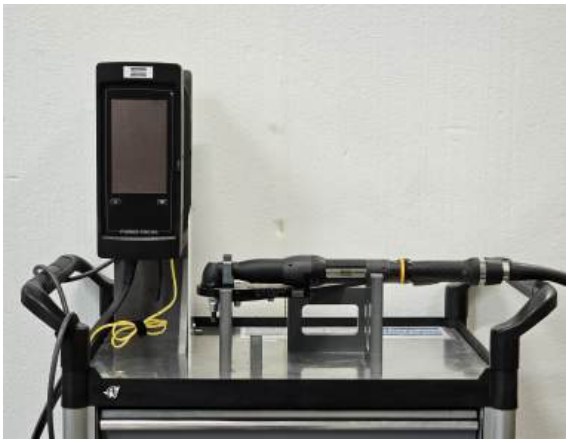

Figure 3: The Atlas Copco nutrunner ETV STR61-70-13 and Powerfocus 6000.

An industrial electric nutrunner from Atlas Copco shown in Figure 3 will be used in this project. It consists of two units, the first one being the controller and power supply ”Powerfocus 6000” (Atlas Copco, 2018). The second unit is the tool itself, which is ”ETV STR61-70-13” (Atlas Copco, 2021). The tool is an angled nutrunner capable of tightening bolts to a torque of 15-80 N m. The tool has a built-in transducer that records the torque and an encoder that records the angle during tightening.

The controller is equipped with a touch screen, which enables the user to change the tightening parameters and view basic results of performed tightenings. Through this interface, basic tightening programs called ”quick step” can be set up with target torques and tightening angles.

3.2.1 Multi-step Programs

To access the full capabilities of the controller, a computer can be connected to access the interface through a web browser. This allows the programming of ”multi-step” programs where multiple steps can be combined and the parameters for each step can be controlled, such as speeds, target torques, and/or tightening and untightening angles. An example of a multi-step program on the Powerfocus 6000 is presented in Figure 4.

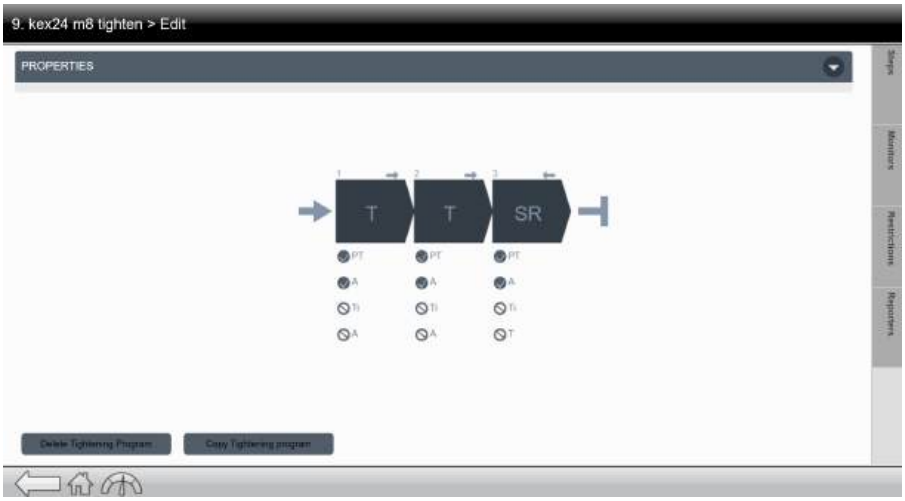

Figure 4: A multi-step program using two ”tighten to angle” and one ”socket release” block.

Here, the multi-step blocks used during experiments are described, and the significant parameters are explained. For all blocks, there are options to add limits for angle or torque to avoid overshooting the limits of the screw. The blocks have many functions that are not used and will therefore not be explained further.

Tighten/Loosen to Torque: This block will tighten the screw to a specific torque value. This can be used to achieve a final tightening torque on the bolt but also as a way of making sure that the bolt is turned to the snug point. Doing this will also make sure that any play in the specimen or socket has been reduced. Depending on the desired use case, it is important to note that the ”Loosen to torque” block by default will only stop when it detects the torque decreasing to the target torque. This can be changed by checking the ”Stop on first torque” switch inside the options of the block.

Tighten/Loosen to Angle: This block will turn the screw a certain angle. This can be used to achieve a certain tightening angle. It can also be used to loosen the screw by turning it a greater angle than the required tightening angle for said screw.

TTTR (Tighten To Trigger Release) This block will tighten the bolt using a specified speed until the operator releases the trigger of the tool. This can be used to tighten the

bolt to a point that is observed by the operator. This block is limited by the fact that it can only be used while turning the screw clockwise.

Socket Release This block rotates the socket a couple of degrees counterclockwise to unload it. This makes it easier to disengage the tool from the screw after tightening.

3.2.2 Fixture for specimen and tool

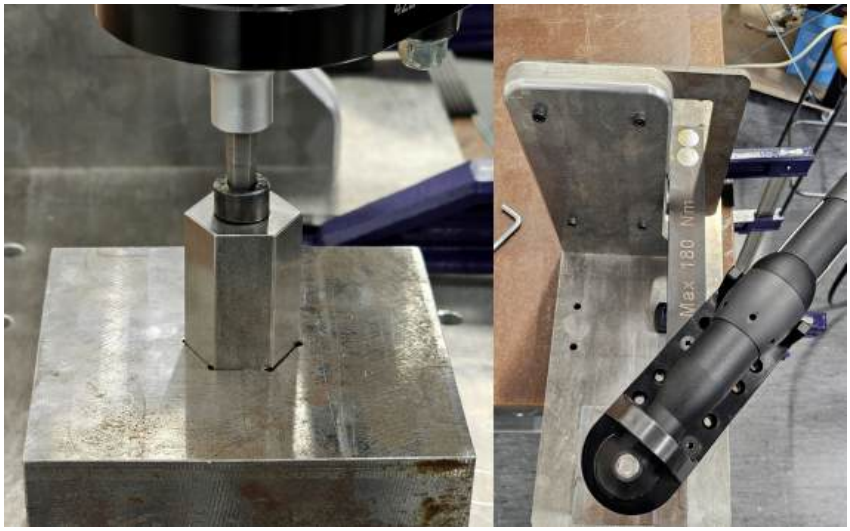

Figure 5: The fixture for the specimen and nutrunner. Left: The specimen placed in the hexagon hole. Right: The reaction plate of the tool positioned in its slot.

When using the nutrunner for experiments, the fixture in Figure 5 was used. The purpose of it is to minimize the influence the user has on the operation. The specimen is placed in a hexagonal hole which will hold the specimen firmly during tightening and untightening. The fixture also has a rectangular slot in which the reaction plate on the tool can be placed. This limits the tool from rotating from the reaction forces during tightening and untightening.

3.3 Impact Drive Rig

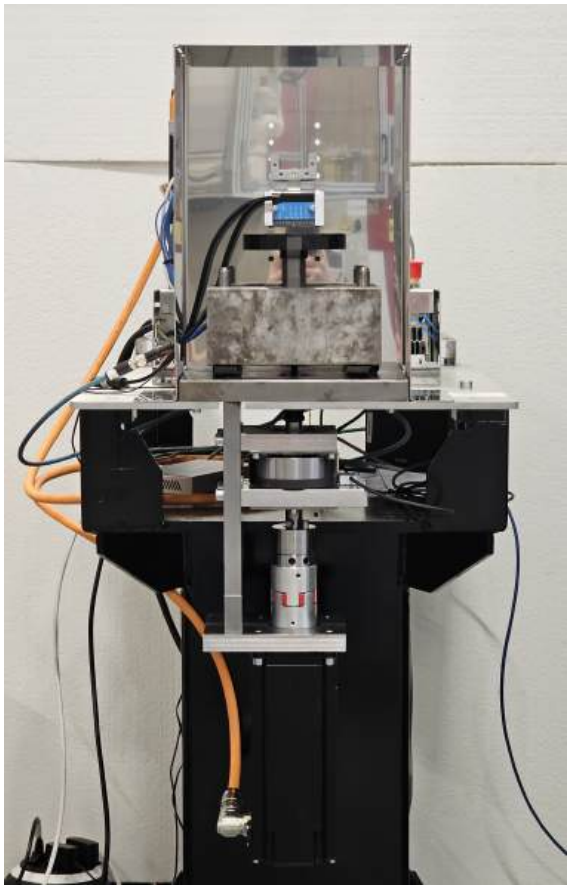

Figure 6: A picture of the impact drive rig in real life.

As stated in the introduction, previous bachelor projects at LiU have developed a test rig to investigate the dynamic properties of an impact drive. The rig is shown in Figure 6 above. In this section, a thorough explanation of how the rig works and what parts it consists of will be laid out.

3.3.1 Motor

The rig is equipped with a B&R 8LSA46.DB060S300-3 three-phase synchronous motor. Details about the motor are available in the datasheet on B&R's product page (B&R, 2024b).

3.3.2 Driver

The B&R ACOPOS 8EI8X8HWS10.0600-1 servo drive controls the motor through its own separate control loop with a sample time of 50 microseconds. On B&R's product page, the datasheet is provided (B&R, 2024a).

3.3.3 Programmable Logic Controller (PLC)

To control the driver and record data, a B&R X20CP1684 programmable logic controller, or PLC, is used. It has a sampling frequency of 800 microseconds. Further technical details can be found in the datasheet from B&R's product page (B&R, 2024c).

3.3.4 Hammer Encoder

To record the hammer angle, an RLS LM10AV000AA30F00 magnetic encoder reads the difference in magnetic field strength of a magnetic strip placed around the hammer's circumference. The strip is made up of small permanent magnets placed in line with a 2 mm gap. The bachelor group responsible for rig improvements installed this sensor; details can be found in their report (Johansson and Kämpe, 2024). The datasheet for the sensor is available for download on RLS' website (RLS, 2023).

3.3.5 Impact Mechanism

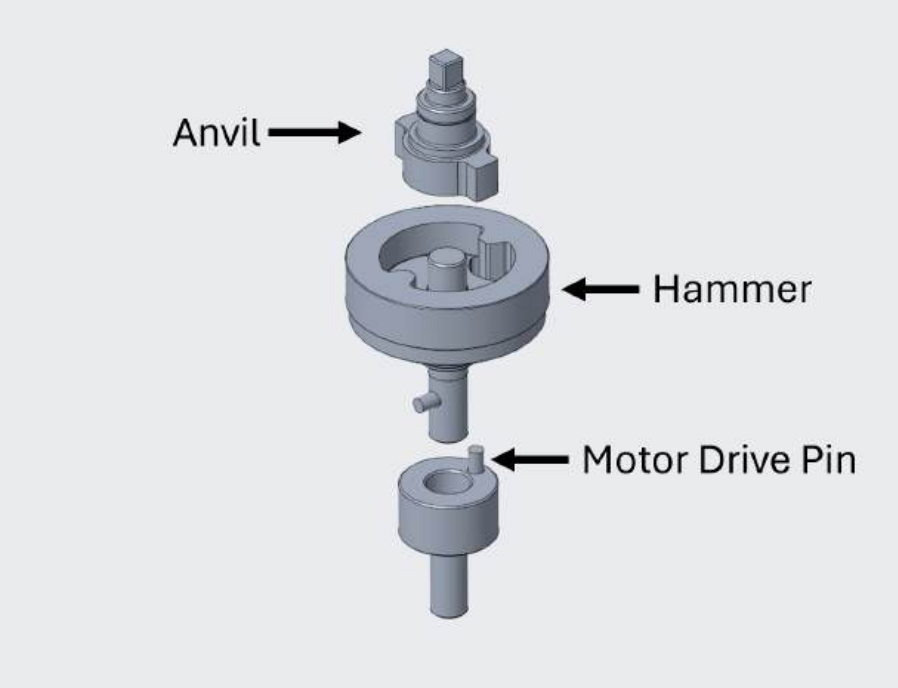

Figure 7: Exploded view of the main parts of the impact mechanism with labels.

Figure 7 shows the main parts of the impact mechanism. Each part and how they interact with each other will be explained below. Details about the design of this assembly can be found in a previous bachelor report (Appeltofft et al., 2023).

Motor Drive Pin: The motor is, via some couplings, connected to the motor drive pin part as seen in Figure 7. This part is responsible for the interaction between the motor and

hammer movement by having the vertical motor pin push against the horizontal hammer pin.

Hammer: The hammer is the part that is accelerated by the motor through the motor drive pin part to hit the anvil at speed, tightening the screw. The value for the hammer's moment of inertia is important and was previously calculated using its CAD model by Appeltofft et al. (2023) to be 0.00284 kgm^2.

Anvil: The anvil, through a square interface at the top, is compatible with different sockets to enable screw tightening. It is the part that gets struck by the hammer, simulating an impact screw tightening.

3.3.6 Specimen holder

Figure 8: The rigs specimen holder.

To hold the specimen in place and at the same time enable it to move up and down when tightening and untightening the screw, a special holder has been designed by Tall et al. (2022). The holder is shown in Figure 8.

3.3.7 Functionality

The rig simulates a impact screw tightening and untightening event through a carefully constructed state machine that runs on the PLC. A custom user interface, shown in Figure 9, has been designed to control the rig.

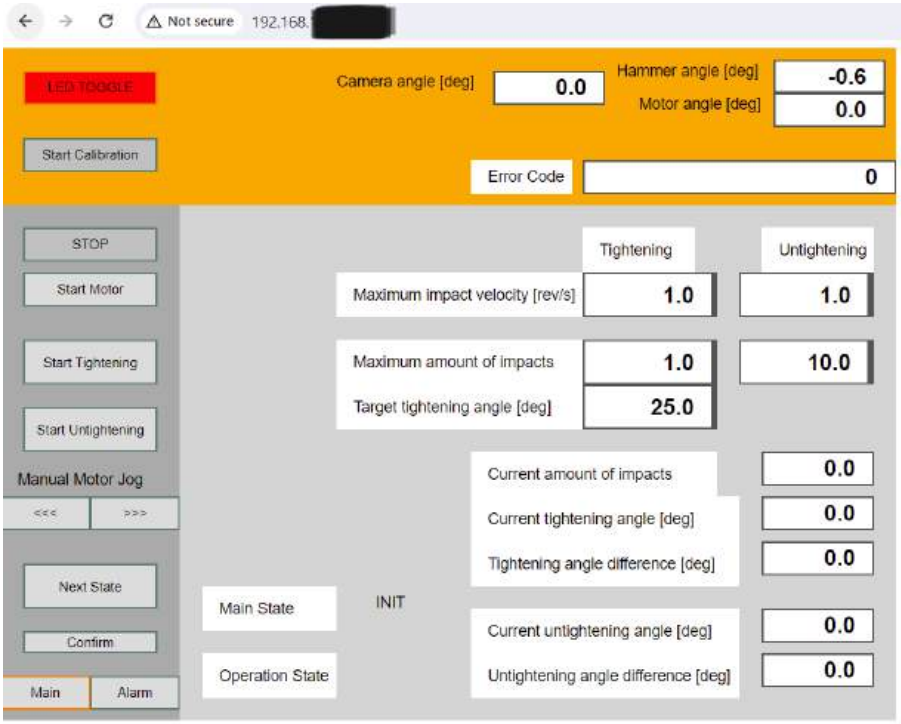

Figure 9: The rigs user interface.

To connect to the UI, the user simply has to type in the rig's local IP address with the correct port into a web browser. The user then decides if they want to tighten or untighten the screw and with what amount of hits and at what speed. A homing procedure sets the zero position for the following tightening or untightening. The next step is repositioning all the main parts. The motor reverses to move the hammer away from the anvil to set up for a hit. Then the motor spins in the opposite direction to make contact with the driving pin and hammer. After the user, through the user interface, confirms that the hammer and motor pins are in contact, the impact forward event begins. In about 0.01 seconds, the motor accelerates the hammer up to the desired target speed. Before the hammer hits the anvil, the motor reverses to avoid getting hit by the hammer when it bounces back from the anvil. The hammer then hits the anvil, which tightens the screw.

3.4 SICK Vision Camera

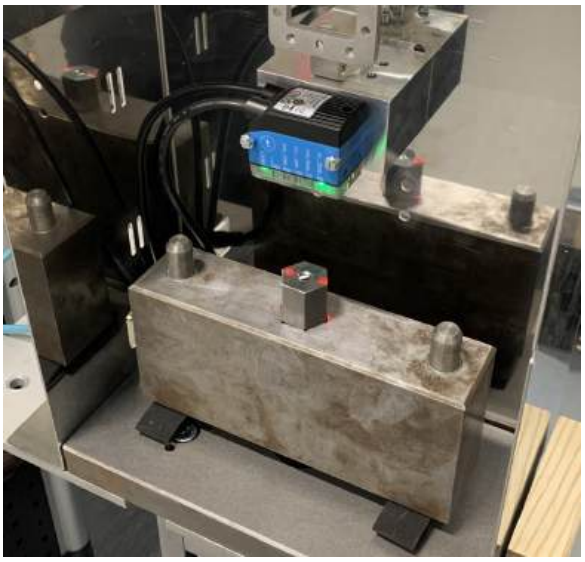

Figure 10: SICK InspectorP61x camera in its holder recording a screw specimen.

A special camera called the V2D611P-MMSBI4 InspectorP61x was kindly provided by the company SICK Sensor Intelligence. It has an integrated machine vision system, making it possible to configure it to find different shapes and lines. To get the best readings possible, a custom-made holder showcased in Figure 10 was built by Johansson and Kämpe (2024), the rig improvement team. For further technical details about the camera, please refer to the datasheet provided by SICK Sensor Intelligence (2024).

The specific use case in this project was to compare the angle between the screw tip and the specimen. This was achieved by designing an enclosure to limit light interference and using 3D printed fittings with high contrast lines. Details about the design process and an analysis of the resulting accuracy can be found in the report written by the rig improvement group (Johansson and Kämpe, 2024).

3.5 Ultrasonic measurement device

Figure 11: The Panametrics-NDT 35 ultrasonic device with its transducer attached to the head of a hex head bolt.

An ultrasonic thickness gauge was provided by Nils Dressler from Atlas Copco. It is a Panametrics-NDT 35 and can be seen in Figure 11. The instrument consists of a hand unit and a transducer with a magnetic tip that can be attached to the object being measured. There is no knowledge about how the device is calibrated since the device is only being used for internal training. It will therefore be tested, and the results will have to be validated and analyzed regarding their validity. Due to only having access to the Panametrics-NDT for two days, the number of experiments and tests that could be done was greatly limited.

3.6 Screw Length Measurement Rig

Figure 12: The screw length measurement rig with the Mitutoyo Digimatic 543-700b attached and a specimen placed for measuring.

A measurement rig purpose-built for measuring the difference in screw length before and after tightening was designed by Johansson and Kämpe (2024) from the rig improvement group in the project. The final rig is shown in Figure 12 above. The rig uses a Mitutoyo Digimatic 543-700b dial gauge with a resolution of 0.0005 mm, which was provided by Mitutoyo (Mitutoyo, 2022).

The purpose of the rig is to hold the specimen steadily to make it possible to get reproducible measurements. When measuring, the bottom of the hex bit interface on the screw head is used as a reference because it is a fairly flat surface. This is done by having the entire specimen resting on a spike that reaches this flat surface. The dial gauge then measures at the tip of the threaded part of the screw. The screws used with the measurement rig had to be flattened where the dial gauge is in contact due to the irregularities on the tip of the screw.

Because the measurement rig was completed late in the project, the extent to which it could be used was limited. For more information and details about the construction, see the other group's report (Johansson and Kämpe, 2024).

3.7 Software

This section provides a short presentation of the different programs that were used for the project.

3.7.1 Git and GitLab

Git is a version control system that tracks changes in source code, enabling collaboration among developers by managing branches, merges, and version history (Git, 2024). GitLab is an implementation of Git that utilises it for repository management, issue tracking, and continuous integration/deployment (Gitlab, 2024). It offers a hub for teams to collaborate and manage projects efficiently. In this project, GitLab was leveraged for two main reasons. Firstly, it was used as a centralised shared file storage where all the project files were included, spanning everything from documents and plots to PLC code and OpenModelica models. The other use case was for setting up milestones for each sprint meeting where all the relevant tasks would be included as issues, ensuring all group members knew what to do.

3.7.2 B&R Automation Studio

B&R Automation Studio is a purpose-built platform for integration and programming of automation hardware (B&R, 2024d). In this project, it is used as the primary Integrated Development Environment or IDE to program the rig's PLC and integrate new hardware components such as the SICK camera or the new hammer encoder. Automation Studio offers several different languages and ways to write code with everything from pure ANSI C to ladder diagrams. Structured Text or ST is the preferred language used in this project to write all the automation logic mainly because of its simplicity.

3.7.3 Python and Jupyter Notebook

The data analysis for this project is mainly done in Python using the Jupyter Notebook format. Python is a high-level interpreted programming language perfect for scientific data analysis with its many libraries (Python, 2024). Jupyter Notebook is a dynamic way to implement Python code where code cells are utilised to run different code snippets separately (Jupyter, 2024).

3.7.4 Matlab and Simulink

Matlab is a high-level programming language and interactive environment primarily used for numerical computing (Mathworks, 2024b). It provides a range of built-in functions and toolboxes for data analysis, visualisation, and mathematical modelling. Simulink is a graphical simulation and model-based design environment integrated with Matlab (Mathworks, 2024a). It allows for simulations and analysis of dynamic systems using block diagrams and visual representations. In this project, Matlab was used for making calculations from the obtained data. The way that the data itself was extracted during test runs with the rig was through a Simulink program with B&R's integrated Simulink blocks.

4 Theory

In this chapter, theory and expressions required to understand the rest of the report will be presented. This theory heavily relies on fundamental theory of screw mechanics found in the book ”Introduction to the Design and Behavior of Bolted Joints, Non-Gasketed Joints” (Bickford, 2008). As this is a continuation of the work of previous bachelor theses, it will also build upon findings and insights from their reports.

4.1 Screw Joint

The screw joint comprises of three main parts: the screw, the clamped parts, and a threaded interface for the screw to interact with. The threaded interface can be either a nut or a thread embedded into the clamped parts.

4.1.1 Screw Tightening

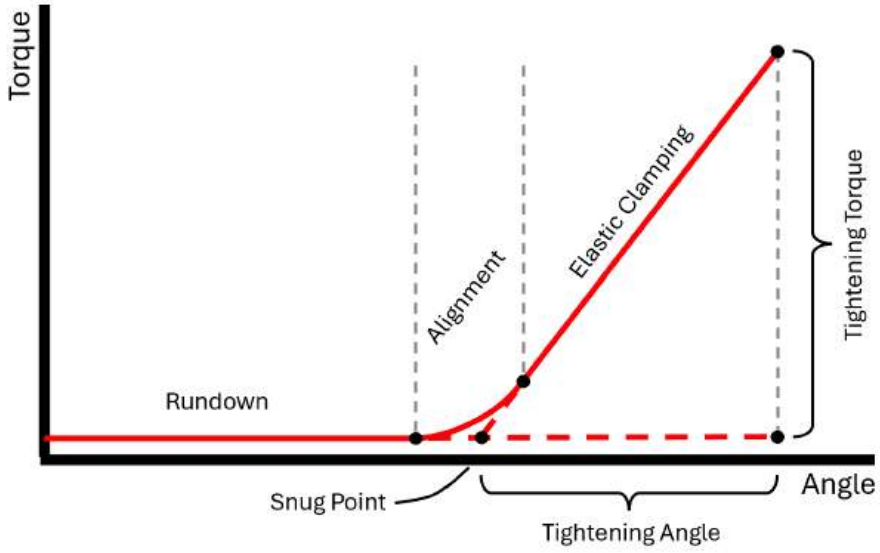

Figure 13: The tightening process based on the description from Shoberg (2000).

The screw tightening process is shown in Figure 13. The first part of tightening the screw is rundown where the screw is turned without the build-up of preload in the joint (Shoberg, 2000). The second part is alignment as the screw head starts to engage with the surface of the joint member. During this, there will be an onset of torque as the head gets aligned with the joint member’s surface. As the screw is further tightened, the torque will increase linearly with the tightening angle. From this part of the tightening process, a line can be fitted. The ”Snug point” can then be found by following it down to the rundown torque. It is from this point the tightening angle is measured.

4.2 Screw Mechanics

4.2.1 Threaded bolt Stiffness

The stiffness of the threaded part of a bolt can be calculated using the following equation (IEI/Maskinkonstruktion, 2017):

$$K = \frac{AE}{L} \tag{1}$$

where L is the clamped length of the threaded part of the bolt and E is its Young's modulus. A is the mean area of the threaded part which in turn can be determined with the expression below (IEI/Maskinkonstruktion, 2017):

$$A_{\text{threaded}} = \frac{\pi}{16}(d_1 + d_2)^2 \tag{2}$$

where d_1 is the inner diameter of the threads and d_2 is the mean diameter of the threads.

4.2.2 Relation Between Torque and Preload

There are multiple equations that aim to describe the relationship between the desired preload in the bolt and the required torque $T_{\text{tightening}}$ to be applied. This report relies on the following equation (Bickford, 2008):

$$T_{\text{tightening}} = F_{\text{P}} \left(\frac{P}{2\pi} + \frac{\mu_{\text{t}} r_{\text{t}}}{\cos\beta} + \mu_{\text{n}} r_{\text{n}} \right) \tag{3}$$

Equation 3 is a simplification which results in a small error on the results (Bickford, 2008). However, the formulation presents the three major components that contribute to the required torque in a clear way. The first of the three components is:

$$F_{\text{P}} \frac{P}{2\pi} \quad ,$$

where P is the pitch of the thread. This component is produced by the inclined plane between the engaged threads. This component of the torque is the one responsible for stretching the bolt. The second one:

$$F_{\text{P}} \frac{\mu_{\text{t}} r_{\text{t}}}{\cos\beta} \quad ,$$

is produced by the friction in the thread. Where μ_{t} is the friction constant in the thread, r_{t} is the mean radius of the thread, and β is the half-angle of the thread. The third component:

$$F_{\text{P}} \mu_{\text{n}} r_{\text{n}}$$

is produced by the friction under the head of the bolt. Where μ_n is the friction constant between the screw head and flange, r_n is the mean radius of the contact area between the screw head and flange. This equation is heavily dependent on the friction coefficient in the thread and under the head of the bolt. According to Bickford (2008), there are 30-40 different variables that affect the friction the fastener is subject to during tightening. The friction is affected by the hardness of the parts, the surface finishes, the presence and state of any lubricant in the joint, just to name a few. Because of this, it is difficult to get accurate estimations of the required torque and or achieved preload.

By changing sign on the first term in equation 3 which is responsible for the stretch of the bolt, an equation for the untightening torque can be formulated:

$$T_{\text{untightening}} = F_P \left(-\frac{P}{2\pi} + \frac{\mu_t r_t}{\cos\beta} + \mu_n r_n \right) \quad . \tag{4}$$

4.2.3 Relation between bolt stretch and preload

To avoid the multitude of uncertainties connected to friction, the preload in the bolt can be calculated using the elongation ΔL. The preload present in the joint can then be calculated using the following equation (Bickford, 2008):

$$F_P = K_B \cdot \Delta L \quad , \tag{5}$$

with the bolt stiffness K_B.

4.2.4 Relation between angle and preload

Using the geometry of the screw, the total deformation in the joint, ΔL, can be calculated using the following equation from Bickford (2008):

$$\Delta L = P\frac{\theta_R}{360} \quad , \tag{6}$$

from the pitch, P, of the thread and the angle, θ_R, the screw has been rotated from the snug point. The previous equation for the preload Equation 5 can be used to estimate the preload by plugging in the deformation ΔL from Equation 6 above and changing the stiffness from the bolt stiffness, K_B, to the joint stiffness, K_J. This yields the following equation for the preload:

$$F_P = K_J P\frac{\theta_R}{360} \quad . \tag{7}$$

4.3 Mechanics

The rotational kinetic energy $E_{\text{rotational}}$ of a rotating object can be calculated using the following equation (IEI/Maskinkonstruktion, 2017):

$$E_{\text{rotational}} = \frac{I\omega^2}{2} \tag{8}$$

where I is the rotational inertia of the object and ω is the rotational velocity.

The potential energy U_{torsion} stored in a torsion spring that has been twisted by θ_{twist} degrees can be calculated from the following equation (Olsson, 2006):

$$U_{\text{torsion}} = \frac{K_{\text{torsion}}\theta_{\text{twist}}^2}{2} \quad , \tag{9}$$

where K_{torsion} is the torsional stiffness of the twisted object.

The torsional torque created by an object twisted to an angle of θ_{twist} degrees can be calculated using the equation (Alfredsson, 2014):

$$T = K_{\text{torsion}}\theta_{\text{twist}} \quad . \tag{10}$$

The torsional stiffness K_{torsion} for a beam with a constant cross section and the length L can be calculated as (Alfredsson, 2014):

$$K_{\text{torsion}} = \frac{GJ}{L} \quad , \tag{11}$$

where G is the shear modulus for the material and J is the polar moment of inertia of the cross section, which for a circular cross section with the diameter D is given as (Alfredsson, 2014):

$$J_{\text{circle}} = \frac{\pi D^4}{32} \quad . \tag{12}$$

With multiple springs acting together in series, the combined stiffness is calculated as (IEI/Maskinkonstruktion, 2017):

$$K_{\text{series}} = \left(\frac{1}{K_1} + ... + \frac{1}{K_{\text{n}}}\right)^{-1} \quad . \tag{13}$$

5 Categorisation

This chapter presents the different categories that the estimation methods will be categorised into.

5.1 Destructive vs Non-Destructive

This category specifies whether or not the method in question has to interfere with the screw joint in order to estimate the preload. Interfering meaning that the method in some way either temporarily or permanently causes a change in preload.

5.2 During vs Post Tightening

For this category, a comparison between when the estimation method can be executed will be made. For example, a method that can estimate the achieved preload while tightening the screw would work during tightening. A method that, on the other hand, can be used whenever, meaning that one can come back to a bolt a week or a year later and check how much preload is left, falls into the other category.

5.3 Generally Usable vs Usable on the Rig

This section is meant to separate methods that are practical to implement in real-life situations. This is a needed distinction since some cases work fine in a controlled lab setup but fail in repeatability and other factors outside of the lab environment.

6 Estimation methods

6.1 Direct Preload Measurement

Direct preload estimation can be made by using off-the-shelf components such as load cells or strain-gauged bolts, which measure the preload in the joint. There are also methods which use special washers and bolts that indicate when the correct preload has been achieved.

6.1.1 Strain-Gauged Bolt

Strain-gauged bolts are specially designed with an embedded strain sensor (Bickford, 2008). These often have a connector in the head of the screw, allowing an operator to read the strain from the sensor which can then be translated to a preload in the bolt. These bolts come with a considerably higher per-unit cost compared to normal bolts, which makes them unsuitable for constructions with a large number of elements. This method cannot be used during tightening as it requires a cable to be connected to the top of the bolt. However, it makes it possible to check the preload after tightening, as well as going back after some time has passed to check the remaining preload. This type of bolt is not applicable for estimating preload in our rig, as the aim of this work is to find more general ways of estimating preload without making changes to the bolted joint.

6.1.2 Load Cell

By placing a load cell somewhere between the flanges of the fastener, one can read a direct force value and thereby get the preload directly from the sensor. This method is classified as destructive since it changes the characteristics of the system. By placing the sensor between the flanges, the stiffness and friction of the joint is affected which changes the final preload value. This fact makes the method in question only suitable for comparing tightening parameters' effects on the preload, since not much is said about the actual preload. This alone makes the load cell obsolete for testing on our rig. For the specific use case of making the aforementioned comparisons, however, this method is usable.

6.2 Preload Estimation by Stretch

These methods rely on measuring the stretch in the bolt. Using this measurement, the preload in the joint can then be estimated with Equation 5. The quality of the estimation is determined by the accuracy of the measurement and knowledge about the stiffness of the bolt being examined.

6.2.1 Micrometer

A previous bachelor thesis at LiU explores the possibility of using a micrometer to measure the elongation of the bolt before and after tightening to determine the preload (Eriksson et al., 2022). They found that their equipment did not have a high enough accuracy to achieve a satisfactory result. This method will therefore not be investigated further.

6.2.2 Dial Gauge

The digital dial gauge provided by Mitutoyo and the screw length measurement rig can be used to acquire the difference in length needed to calculate the preload using Equation 5. The dial gauge, with its high accuracy and a rig that enables greater consistency than what was previously achievable using the micrometer, should yield more reliable results.

To estimate the preload, the dial gauge should be zeroed before tightening the screw. The preload can then be estimated by measuring the change in length after tightening. For an already tightened screw joint, the dial gauge can be zeroed on the tightened screw. The change in length can then be measured after loosening.

When using the measurement rig, it was found that the specimen is not perfectly held in place. The geometry of the rig allows some play in the positioning of the specimen. The impact of this varied from test to test. To minimize the effect this would have on the result, the specimen was always placed in the same way into the rig.

This method is feasible for our experimental setup, which only has one specific specimen, making it easier to manufacture a structure to repeatably measure the length. However, this method would be hard to implement on a general screw joint. In many cases, both sides of the screw are not accessible to measure the length of the screw, rendering this method unusable. Additionally, for an already tightened bolt, this method also requires the screw to be completely untightened to estimate the preload that was present.

6.2.3 Ultrasonic Measurement Device

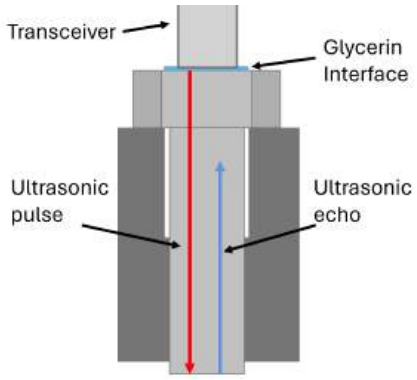

Figure 14: The transceiver emits an ultrasonic pulse that is reflected as an echo of the bottom of the screw.

Ultrasonic measurement of the screw stretch works by placing an ultrasonic transceiver that is connected to an electronic device on one end of the screw (Bickford, 2008). The transceiver sends out an ultrasonic pulse or burst that travels through the bolt, as illustrated in Figure 14. The pulse then echoes off the other side and is received by the transceiver. The electronic device will then very accurately measure the time, the transit time, between when the pulse was sent and the echo is received.

According to Bickford (2008), there are two factors that contribute to the time between the pulse and echo to increase while tightening a screw. The first contributor is the stretch of the bolt, leading to an increased path for the pulse to travel. The second comes from the fact that the speed of sound will decrease when the stress in the material is increased. Both of these are linear functions of the preload which means that the transit time is also a linear function of preload.

When using ultrasonic sensor to measure screw stretch, it is important to calibrate the equipment for each use case (Bickford, 2008). This is because the speed of sound in various materials and the change due to stress greatly differs between different materials. There can even be differences when testing what should be the same material from different lots.

A special screw with a flattened head was used with the ultrasonic measurement equipment. The measurement probe is dependant on a flat surface in order to get good contact. Glycerin is applied as an interface between the screw and the probe, the application can be seen in Figure 15 below.

Figure 15: Glycerin applied to the head of the prepared hex head screw.

6.3 Preload Estimation by Angle

The following methods utilise Equation 7 by measuring the tightening angle of the bolt in different ways and then calculating the preload using the stiffness of the joint. These methods both rely on having an accurate angle measurement and an accurate estimation of the stiffness.

6.3.1 Impact Drive Rig

The rig measures the tightening angle at the screw head after every hit. This is done by first zeroing the angle when the screw is snug and then recording the relative difference between this zero and the new tightening angle after a hit. This angle is influenced by several external factors. First of all, the snug point could vary between tightenings. This would mainly occur if there is a difference in rundown distance and therefore hammer speed, since the inertia of the hammer would pre-tighten the screw by different amounts. This factor could be avoided in most tests by ensuring the screw was close to snug before placing it in the rig. More influential factors such as variation in friction from hit to hit or wear down of the screw were harder to quantify and avoid. Assuming the measured angle is to be trusted and using the stiffness calculated by Augustini et al. (2022), an estimation of the preload is possible.

6.3.2 SICK Camera

Using the SICK camera with its integrated machine vision software would provide another way of obtaining a measurement of the tightening angle, specifically on the tip of the screw.

6.4 Preload Estimation by Torque

The below listed methods estimates the preload by using Equation 3 from Bickford (2008). Solving said equation for the preload F_P gives the following equation:

$$F_{\mathrm{P}} = \frac{\mathrm{T}}{\frac{P}{2\pi} + \frac{\mu_{\mathrm{t}} r_{\mathrm{t}}}{cos\beta} + \mu_{\mathrm{n}} r_{\mathrm{n}}} \quad . \tag{14}$$

Knowing the applied torque and screw parameters, the only unknown parameter is the friction in the thread and under the head of the bolt.

6.4.1 Atlas Copco Nutrunner - Retightening Torque

One way of using the nutrunner to estimate the preload of an already tightened joint is to loosen it and then turn it to the same position. The retightening torque of the joint can then be obtained. Before loosening, the screw head and flange need to be indexed

using two pieces of tape, as shown in Figure 16. This is done to avoid the influence of the stiffness between the screw head and the encoder on the nutrunner when attempting to tighten the screw to the same position.

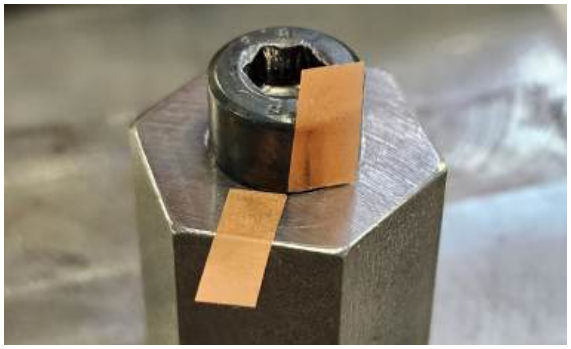

Figure 16: Two pieces of tape placed on the flange and screw head to mark the tightening position.

The method is performed using a multi-step program that first uses a ”tighten to torque” block set at 1 Nm with a high rotational speed to run the screw down until it is snug against the flange. Then, a ”Tighten to trigger release” block is used at a low rotational speed to enable releasing the trigger when the two pieces of tape line up. The required retightening torque can then be extracted and used together with Equation 14 to calculate the preload.

6.5 Tighening-Untightening Torque

By subtracting Equation 4 from Equation 3 and solving for the preload F_P a new equation is formulated:

$$F_\mathrm{P} = \frac{T_\mathrm{tightening} - T_\mathrm{untightening}}{P} \cdot \pi \tag{15}$$

which gives the preload as a function of the difference in tightening and untightening torque and the pitch of the thread. This equation is independent of the friction in the joint. However, it assumes that all conditions are the same when determining the tightening and untightening torque. This method can be performed on a already tightened screw joint. It is not able to be used during the tightening of the screw.

6.5.1 Atlas Copco Nutrunner

This method can be performed with the nutrunner using a multistep program that includes a ”Tighten to trigger release” block. Initially, the screw is slightly tightened to ensure that the head of the screw rotates a slight angle. Subsequently, another program is executed with a ”Loosen to angle” block. This program fully untightens the screw in our case but could be set to only loosen it a certain angle if desired. The torque required to further

tighten the bolt from its initial position can then be extracted from the trace data from the nutrunner. The rotation of the screw from the ”Tighten to trigger release” step is then taken into account when extracting the untightening torque. It is important to retrieve the difference in torque from the same position. These values can then be used to calculate the preload present in the screw joint using Equation 4.

6.6 Preload Estimation by Inertia

An estimation of the torque applied to a screw from an impact mechanism can be made using the recoil observed after an impact. Recoil refers to the speed at which the hammer ”bounces” off the anvil after striking it. This phenomenon occurs because the interface between the hammer and screw acts as a torsional spring. During the impact, the rotational energy in the hammer transfers through the anvil and socket into the screw, causing it to turn and tighten, resulting in torque acting on the anvil and socket. This torque twists the socket and anvil, storing energy that is later released back into the hammer as recoil.

Assuming that all the stored spring energy in the anvil and socket is released to the hammer as rotational energy, the twist angle θ_{twist} can be calculated using Equation 8 and Equation 9 as:

$$\theta_{\text{twist}} = \sqrt{\frac{I_{\text{hammer}}\omega_{\text{recoil}}^2}{K_{\text{torsion}}}} \tag{16}$$

where ω_{recoil} is the hammer's recoil speed, I_{hammer} is its inertia, and K_{torsion} is the torsional stiffness of the anvil and socket. This can then be inserted into Equation 10, which gives the torque applied to the screw from the twist angle θ_{twist}:

$$T_{\text{impact}} = \sqrt{K_{\text{torsion}} I_{\text{hammer}} \omega_{\text{recoil}}^2} \tag{17}$$

The achieved preload can then be estimated using the calculated torque T_{impact} and Equation 14 presented in Section 6.4.

When the screw can no longer be turned by the impact, all of the energy in the hammer will be converted to torsional energy in the socket and anvil. Equation 17 will then give the highest torque that the system can apply to the screw for a given impact velocity ω, inertia of the hammer and stiffness of the anvil and socket.

6.6.1 Impact Drive Rig - During Tightening

The equations above can be utilised to estimate the preload achieved with the impact drive rig during the tightening process. Preload can be calculated after each impact by measuring the angular velocity of the hammer's recoil after the impact.

6.6.2 Impact Drive Rig - Tightening-Untightening

The previously derived Equation 15 from Section 6.5, which utilises the difference in tightening and untightening torque, in combination with the inertia equations in this section, provides another method of estimating preload. This could be implemented with the rig by performing one tightening hit followed by one untightening hit. The respective applied torques can then be calculated using Equation 17, and these values can be inserted into Equation 15.

6.7 Preload Estimation by Screw Twist

This method is based on Equation 10 and relies on obtaining an accurate measurement of the twist of the screw, θ_{twist}. This, combined with a known torsional stiffness, yields a value for the residual torque left in the screw after tightening. This residual torque can be seen as responsible for keeping the screw twisted along its axis. The only friction affecting the residual torque is that in the screw threads, as the head is considered fixed when considering the twist. With this knowledge, Equation 14 (excluding the last head friction component) can be used to estimate the preload.

6.8 Categorising Estimation Methods

In this section the estimation methods presented above are categorised into the categories presented in section 5. The result of the categorisation is presented in Table 1.

Table 1: Categorisation of estimation methods

Category	Destructive vs Non-Destructive		During Vs Post Tightening		Generally Usable vs Usable with Rig	
Method	Destr.	Non-Destr.	During	Post	General	Rig
Strain gauged bolt		×	×	×	×	
Load cell		×	×	×	×	
Stretch - Micrometer		×		×	×	×
Stretch - Dial Gauge		×		×	×	×
Stretch - Ultra Sonic		×	×	×	×	×
Angle - Rig		×	×			×
Angle - SICK Camera		×	×			×
Torque - Nutrunner	×		×	×	×	
Tight.-Untight. Torque - Nutrunner	×			×	×	
Inertia - Rig	×		×	×		×
Tight.-Untight. Torque - Rig	×			×		×
Twist - Rig & SICK Camera		×	×			×

7 Rig Development

To investigate all methods involving the impact drive rig, a substantial amount of time was devoted to further developing its capabilities. This work consisted of both expanding the software through programming the PLC and making hardware improvements. Most of the work in this chapter builds upon the software created by Gustavsson and Erlandsson (2023) for a mechatronics project course.

7.1 Programming the PLC

The software running on the PLC is developed in B&R Automation Studio. The language used in this project was primarily Structured Text (ST). The first step was to create a program for loosening screws on the rig, which originally only tightened them. Relevant code sections are included in the appendix, and the full program is available on Gitlab through contacting the project supervisor, Martin Hochwallner. The program utilises state machines to control the rig.

7.1.1 Code Abstraction

To add a completely new sequence for an untightening procedure, a new state machine had to be made. The code existing in a single main file quickly became overwhelming to work with, so the decision to divide the program into sub-programs, or in B&R's terms, ACTIONS, was made. Two new action.st files were added, one for tightening and one for untightening. The main state machine was kept in the main.st file (found in Appendix A.1), while the other two were moved to their respective action files. This not only helped with readability but also had the added benefit of making the code scalable, an aspect that would come up several times during development.

7.1.2 Untightening Program

Using the already constructed tightening program as a base, a basic untightening motion could be achieved by simply inverting all the distance variables. This worked for setting up the hammer-anvil and striking the screw in the other direction. However, some new logic was needed to handle the moment when the screw became completely loose. After every hit, the untightening angle is checked and recorded while also checking if the screw had loosened on that very hit. This check is made by seeing if the motor can make two full rotations without interruption when trying to record the untightening angle. If these revolutions are possible, the screw is considered completely loose. To then record the final untightening angle, the motor moves the screw in the opposite, tightening direction until hitting the snug point by stopping when the torque reaches 0.3 Nm. This snug position is seen as the starting position and can therefore be compared to the previous untightening

angle to get the final total angle. Both the tightening and untightening programs can be found in Appendix A.2.

7.1.3 Calibration program

Another functionality that was worked on was a calibration program as having all tests start from the exact same position would mean more consistent data and higher accuracy in the results. This calibration sequence is supposed to use the motors absolute encoder to ensure the correct position is reached. The calibration program was not finished due to time restraints.

7.1.4 User interface

During the rig development the user interface had to be updated continually to support new features such as untightening. This was done using Automation Studios visual editor where one can bind specific process variables to blocks in the UI. Examples being the new camera angle to a text block or a start untightening boolean to a toggle button. The final UI is shown in Figure 9 in the Equipment chapter.

7.1.5 Data extraction with Simulink

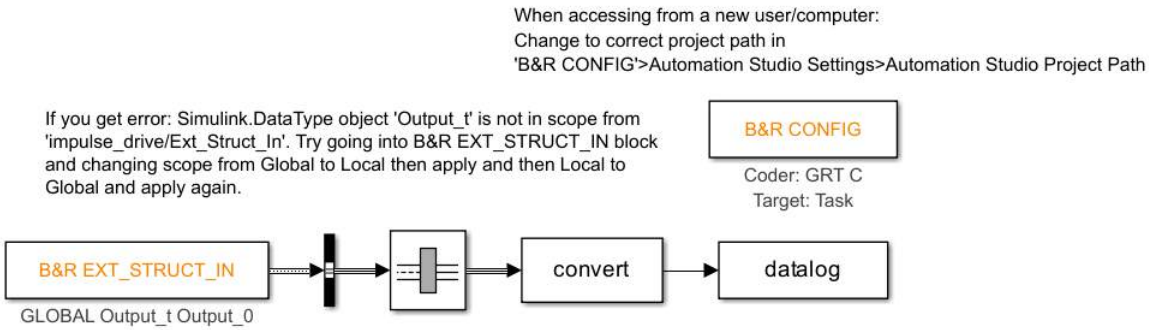

Figure 17: Simulink model

To get useful data recorded during test runs and extracting it to a data analysis software such as Matlab or Python a target for Automation Studio in Simulink had previously been made. By adding so called B&R-in blocks in Simulink and matching the parameter names to the ones being used in the PLC program these values would be recorded into a datalog variable available in the Matlab workspace for further processing. This method worked great but left some things to be desired in the aspect of flexibility and readability. If a new variable was to be added a new block was needed in Simulink etc. It was not clear in the PLC program which variables was being exported since this was completely

determined in the Simulink program. A new method using the Simulink block 'B&R extended struct' in was implemented. This made it possible to add a struct in Automation Studio called output.typ which contains all the variables being sent to Simulink. A new action in the PLC program, write_output was created serving the purpose of copying the corresponding variables into the output struct as well as converting them all into LREAL numbers in order for some of the Simulink blocks to work as intended. The output.typ struct is read in Simulink and sent so a Bus Creator block that dynamically creates a bus depending on the size of the input struct. This means that the specific outputs are decided in the PLC code and the rest of the processing pipeline adjusts accordingly.

Choosing what variables are logged raises some important aspects to think about. The raw sensor values are usually large unreadable numbers that has to be scaled into units that make sense such as degrees or radians. Since some of the constants in the PLC code are in degrees, sensor scaling has to be done live while the rig is running. In general it would usually be best to output the rawest values possible both for reducing CPU load during runs but also to make sure no data is ruined before being logged. An example of the latter being subtracting an offset from an angle measurement but not exporting the offset value making the data less useful in later processing steps. Keeping these things in mind the degree conversion is the only thing being done to the data in the PLC code.

7.1.6 OPCUA Matlab Integration

There are multible ways of recording constant values such as settings parameters. One simple way is to include them in the datalog in Simulink. However this means that a constant value is being saved for every cycle, sometimes up to 600 000 times. A better way would be to make a new settings struct in Automation Studio and save it in a separate workspace variable in Matlab where the parameters are only recorded once. Another method and the one chosen for this project was to use the OPCUA communication protocol to read the constants into MATLAB. The reason for this choice was mostly to explore the capabilities of OPCUA and how to implement it. OPCUA stands for 'Open Platform Communications Unified Architecture' and is, as the name suggests, a generalised way for machines of different kinds to communicate. The full Matlab file with both the OPCUA code and the data logging is available in Appendix C.3.

7.2 Data Analysis in Python

With all the data saved in a .mat file with the HDP5 format it can be opened in Python with the h5py module. Both the settings- and the datalog struct are read in as dictionaries and converted into Pandas Dataframes for easier manipulation. All this happens in a new python file, Preprocessing.py, found in Appendix B.1. That code also adds useful columns for plotting with new units. Examples being the time in seconds, motor angle

in degrees and a state integer to string conversion. The rest of the data analysis is done in a Jupyter notebook that imports all the functions from the Preprocessing.py module. The specific plots are made using matplotlib often by filtering the data to show a specific moment such as the 0.1 second impact event.

7.3 Integrating the SICK Camera

Most of the work involving the SICK camera was made by the other bachelor project group handling general rig improvements (Johansson and Kämpe, 2024). When they had succeeded in configuring the camera so that accurate angle measurements were possible an integration with the PLC program had to be made. The protocol used by the camera to send data is Ethernet/IP and luckily the PLC was equipped with a special Ethernet module. A seemingly simple connection from camera to program could therefore be made. A lot of trial and error went in to extracting the data. First, only integers could be sent and since the angle resolution had to be in the hundreds of a degree range for the specific use case it was crucial that a float/real could be sent instead. After configuring the camera to send the angle as a REAL the data showed up as four 8 bit integer numbers in the PLC program. These numbers represented a single precision floating point number consisting of 32 bits. Converting the four integers into binary and then from binary, using the IEEE 754 standard float number formula where the first bit represents the sign, the following 8 being the exponent bits and the last 23 being the so called mantissa bits, the data could now be fully converted to the real number. Doing these conversions live every 0.8 ms did not seem to be the best solution since a lot of computing power would be used so another alternative was investigated. Looking at the structure of the data being sent from the camera and telling the PLC program how to interpret it by adding so called channels and mapping them to corresponding process variables in the program worked out in the end.

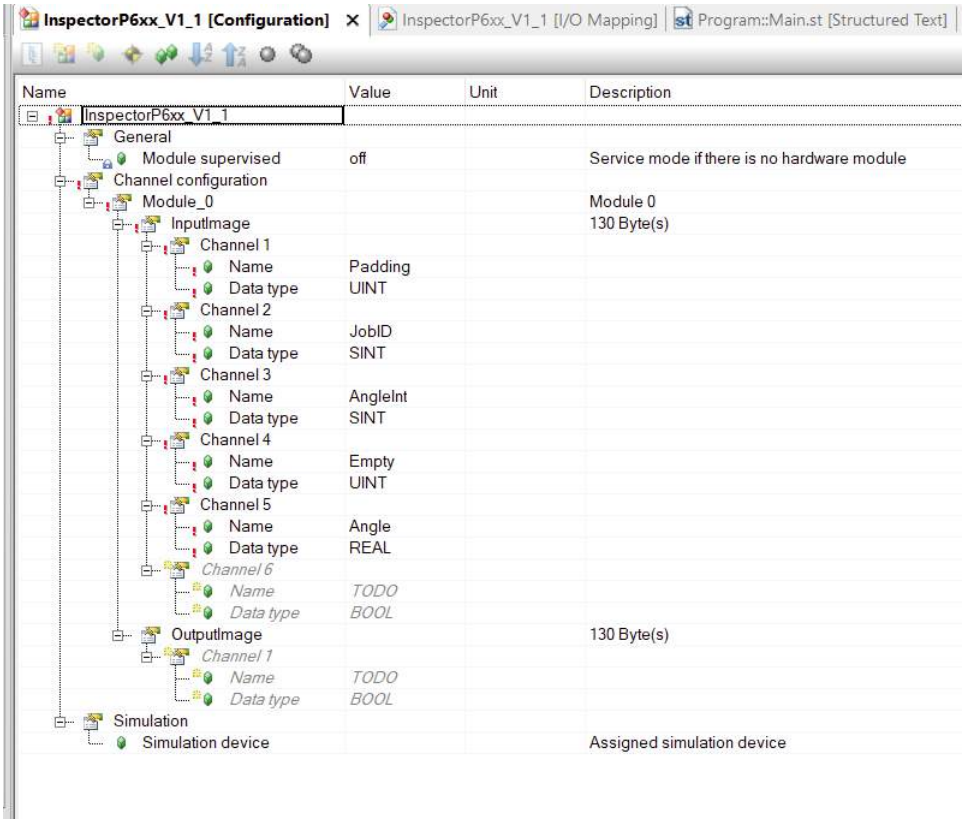

Figure 18: An overview of the different data channels for the SICK camera.

Figure 18 illustrates the channel setup that was implemented to handle the import of data from the camera. The first channel is named ’Padding’ and set to an unsigned integer or UINT intended to read the first two bytes of the data stream. The second channel reads the ’JobID’, a single byte small integer or SINT representing the current camera setup. The third channel gives the angle as another SINT reading the next byte of data with the name ’AngleInt’. The fourth channel is named ’Empty’ since it does not contain useful data, a UINT reads these two empty bytes. The fifth channel is the one to read the angle as a four byte REAL number. By simply setting the data type to REAL in Automation Studio and making sure it is in the correct channel the data interpretation is made automatically by the program.

8 Testing and Analysing Rig Behavior

Some general phenomena of how the different parts of the rig move are consistent in all tests and will be explained and analysed here. Below, Figure 19 displays a full tightening cycle at 5 rev/s with a zoomed-in plot of the impact event.

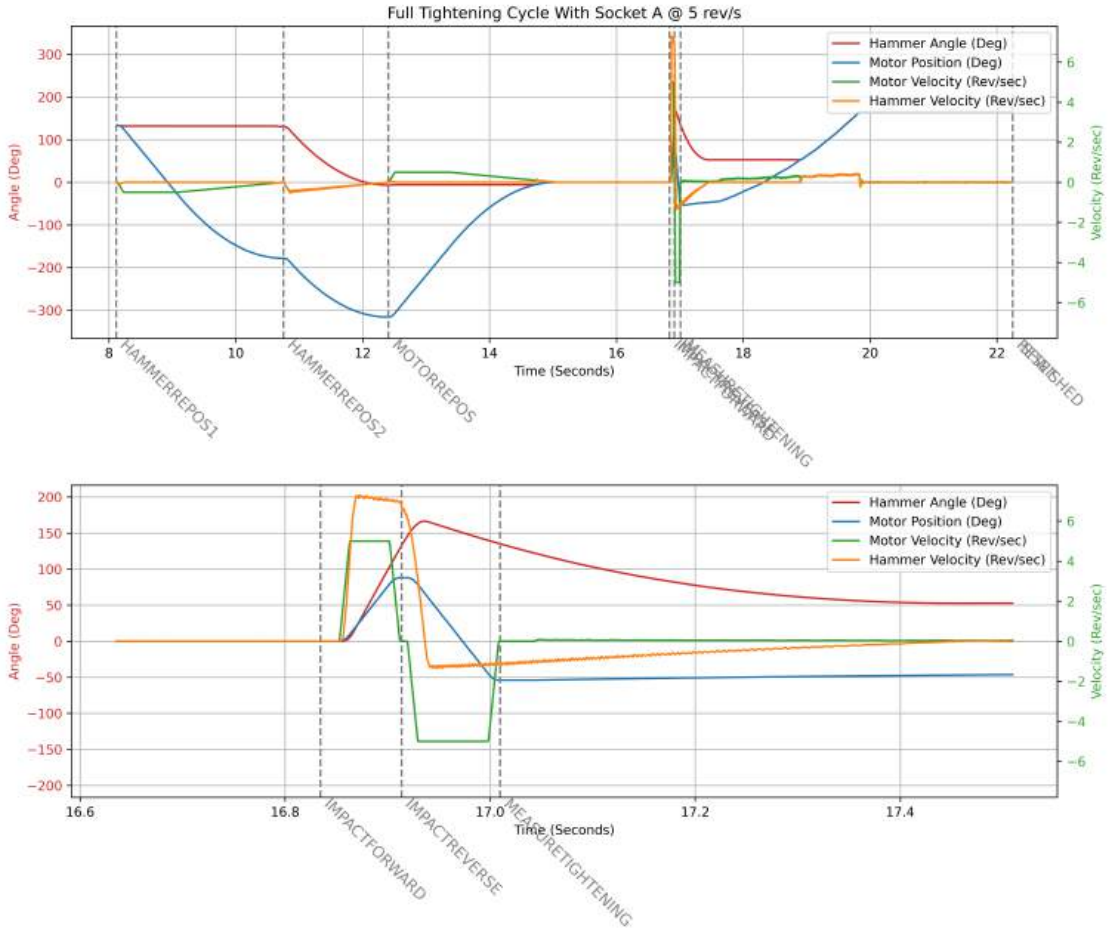

Figure 19: A full tightening cycle with the rig including a zoomed-in plot of the impact event.

8.1 Hammer-Motor Position and Velocity Deviation

Every single tightening and untightening impact shows a noticeable difference in motor and hammer speed, where the hammer accelerates to a much higher velocity than the motor driving it. This can be seen by comparing the green and orange lines in Figure 19.

This behavior likely comes from the pin on the hammer and motor drive pin acting as springs. When the motor pin accelerates the inertia of the hammer, a torque will be applied to the pins which will result in spring energy being stored. When the motor stops accelerating, there will no longer be a torque applied between the drive pin and the hammer. The stored spring energy will then be released, further accelerating the hammer to

a greater speed than the motor speed.

Unfortunately, it was not possible to further investigate this behavior due to there not being enough time. However, it would be interesting to try changing the acceleration of the motor to see the effect this has on this behavior of the rig. Setting a lower acceleration would lead to a lower torque between the motor drive pin and the hammer. This, in turn, would lead to less energy being stored in the spring, and therefore, the hammer should not reach a speed that is much greater than the motor.

8.2 Camera Angle Dip Before Impact

When recording the angle of the tip of the screw with the Sick camera a decrease in angle is seen in every test.

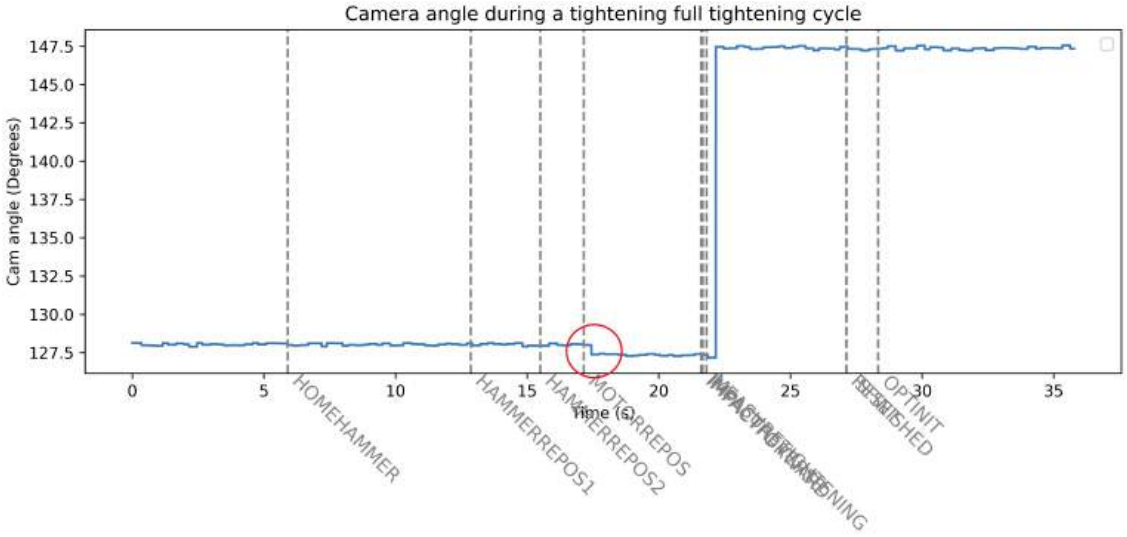

Figure 20: Plot showing the recorded angle from the Sick camera for a full cycle.

Figure 20 is a plot of the recorded angle of the screw tip with the dip mentioned circled in red. The reason this phenomenon occurs could be because the anvil is moved back in the untightening direction because of friction in the bearings when the motor is repositioning the hammer. This is very likely to be the case since the dip in angle always happens right after going into the MOTORREPOS state as seen in Figure 20. To get the best data possible the camera angle before impact is filtered to only include a mean of all values before MOTORREPOS. The mean of the angle from one second after IMPACTFORWARD and to the end of the cycle is seen as the tightening angle at the tip. The reason a mean value is taken is because the camera has some variation in angle over time, this can be seen in Figure 20 where what should be a static value resembles a square wave. The code for processing and plotting the camera angle is provided in Appendix B.2.

8.3 Hammer Velocity Dip Before Impact

In all test runs a relatively small but sharp dip in hammer velocity can be seen right before impact, in Figure 21 this phenomenon is circled. This behavior is unexpected and not included in the OpenModelica model and it was therefore investigated.

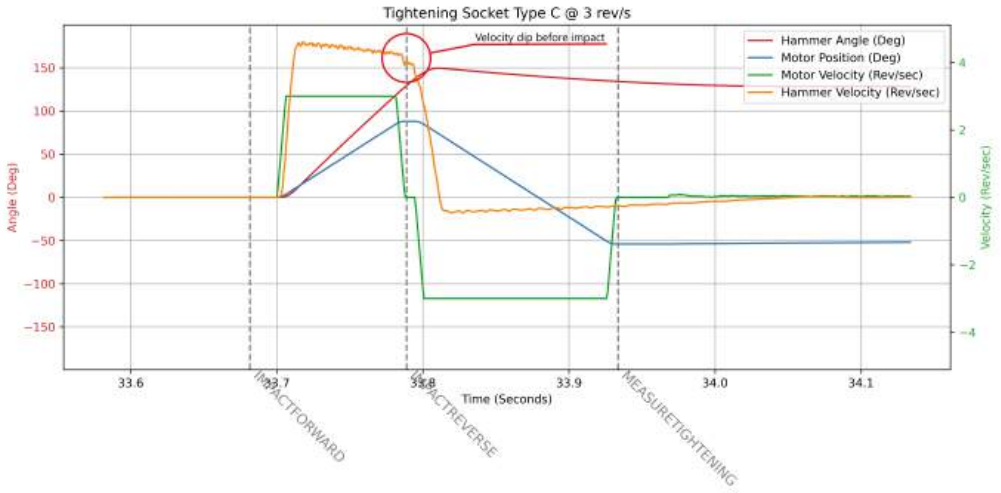

Figure 21: Tightening hit with circled velocity dip.

Motor reversal: The first hypothesis was that the decrease in velocity was caused by the reversal of the motor slowing the hammer down with the friction in the bearings. Moving the point of reversal by decreasing the distance the motor has to cover should make the velocity dip happen earlier. A test where the motor only travels 20 degrees instead of 92 was made and the results are shown below in Figure 22.

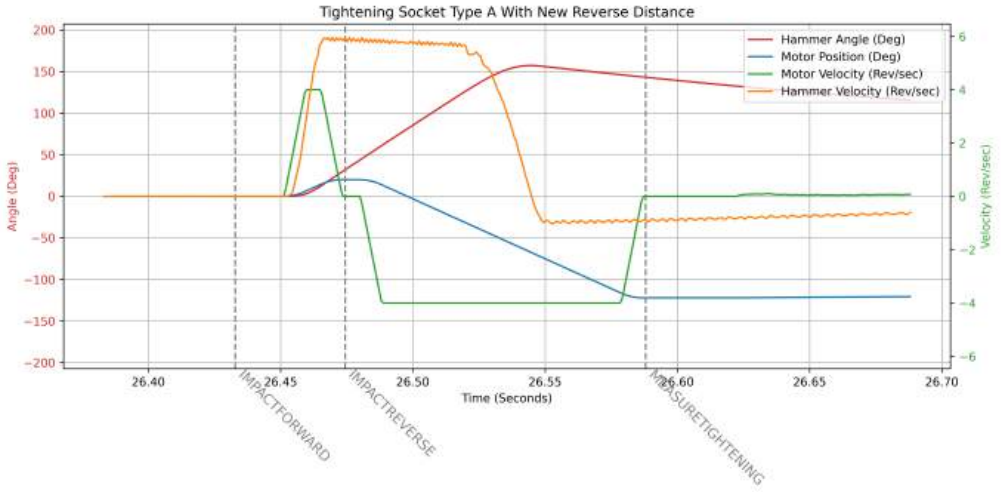

Figure 22: Tightening hit with new motor reverse distance.

This test proves this initial hypothesis to be wrong since the sharp decrease in velocity

happens at the exact time as before even though the motor reverses much earlier in the cycle.

Anvil inertia and friction: A second hypothesis was devised. This time investigating whether or not the dip could be because of there being play in the components between the screw and hammer. The idea being that the anvil with its socket might not be all pressed up against the screw meaning that the hammer first hits the anvil and transfers some of its rotational energy to it. The anvil and hammer will then turn ”freely” for a short bit until it starts to tighten the screw. To test this an experiment was carried out where a thread was used to keep the anvil and socket snug against the screw head, both in the tightening and the untightening direction. This setup is shown in Figure 23 below. The expectation was for the dip in velocity not to be present when the thread was making sure that all parts were snug in the tightening direction. When the thread is pulling in the untightening direction the idea was that the dip would become more prevalent.

Figure 23: Thread test in both the tightening and loosening direction.

The impact with the thread pulling in the tightening direction is shown in Figure 24 below. Contradictory to the expectations the dip in velocity is still present, even though all the play between the hammer and screw head is held together by the thread. This test shows other weird behaviour as Figure 24 shows the motor reaching the target speed while the hammer is still standing still. This is most likely due to the thread causing the anvil to push the hammer. This makes the hammer and motor drive pin not be in contact at the start of the impact. This causes the hammer to reach a lower speed compared to when the thread is pulling in the untightening direction which can be seen in Figure 25 below.

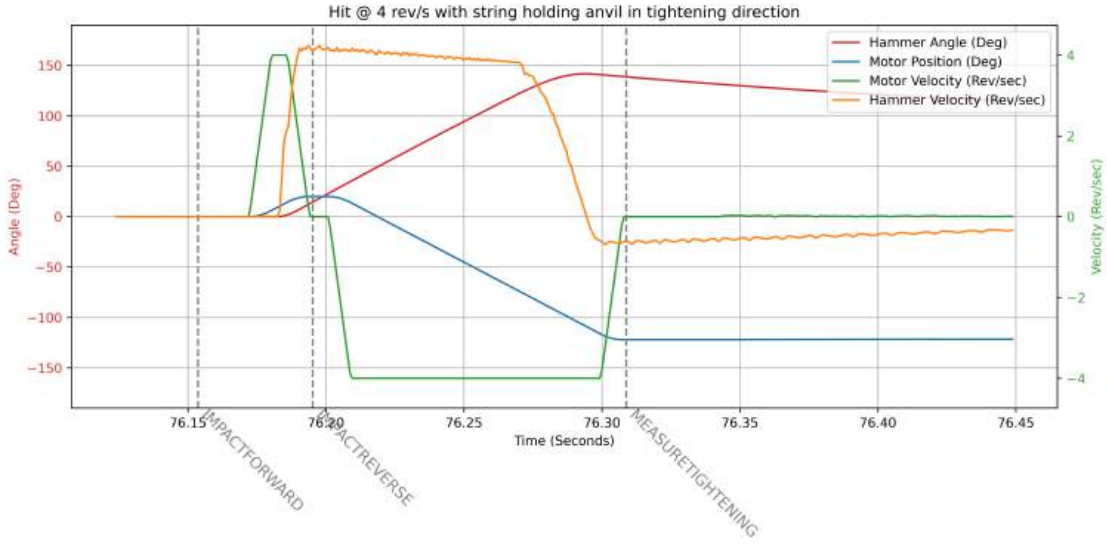

Figure 24: Tightening hit with the thread in the tightening direction.

Figure 25 below shows the test with the thread pulling in the untightening direction, here the dip is not visible in the same way as before. However it instead shows the hammer losing speed at a more gradual pace. This can most likely be contributed to the string stopping the hammer from rotating the anvil making the impact a lot softer which was observed during test.

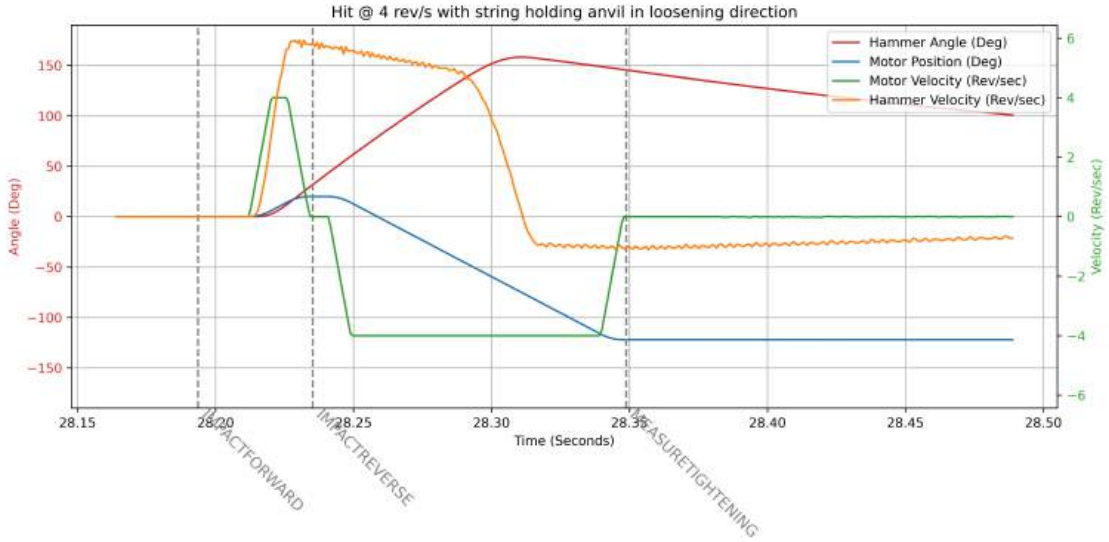

Figure 25: Tightening hit with the thread in the untightening direction.

The experiment points towards the hypothesis about the play in the socket causing the dip being incorrect. The reason for the dip could not be further investigated due to time restraints. There was one suspicion left for the reason behind the dip, this was the fact that the anvil could be bouncing of the hammer. It could be interesting to make a deeper analysis regarding this. Looking at what happens when the screw is tightened with multiple hits and at different speed to see how the behaviour varies.

8.4 Analysing Stiffness Between Hammer and Screw

When tightening the screw using the impact drive the torsional stiffness between the hammer and screw is of great interest. As can be seen in equation 17 the stiffness is one of the variables that control the torque that the hammer can exert on the screw. To determine the stiffness between hammer and screw in our system an experiment where the rig repeatedly hits the screw until it is no longer able to turn it, was performed. This would mean that all rotational energy in the hammer is converted to torsional energy between hammer and screw. Solving equation 17 for the stiffness K_{torsion} gives the expression:

$$K_{\text{torsion}} = \frac{T^2}{I_{\text{hammer}} * \omega_{\text{hammer}}^2} \tag{18}$$

for the torsional stiffness. Where the Inertia is known from previous bachelor thesis. The torque applied from the impacts can be determined using the ”Retightening torque” method described in section 6.4.1. The angular velocity of the hammer at the impact can be determined by analysing the recording from the rig.

Figure 26: The different sockets used to test the stiffness. Type A socket to the left, Type B in center and Type C to the right.

In this Experiment specimen A was used in combination with three different types of sockets shown in Figure 26. The first one, Type A is a 6 mm ”krafthylsa” with a hex bit socket, meant for impact drivers. The second, Type B is a normal 6 mm hex socket bit. The third, Type C is a 13 mm hex socket. For the hex socket a hex head bolt had to be used. The rig was set to tighten the specimen with a motor speed of 2, 2.5 and 3 rev/s for the Type A and Type B sockets. For the Type C socket, tightening was only performed with 2 and 3 rev/s.

The hypothesis was for the stiffness of the Type A socket to be remarkably lower than the stiffness of the Type B socket according to Equation 11 due to its longer and slimmer geometry. The stiffness for the Type C socket was also expected to be greatest of all of the sockets.

8.4.1 Result

The result of the experiment is presented in Table 2. The Matlab file used to calculate the stiffnesses with the values extracted from the trace files from the nutrunner and impact drive rig can be found in Appendix C.1.

Table 2: Stiffness between hammer and screw calculated with equation 18

Socket	Motor Speed [rev/s]			Average Stiffness [N m/rad]
	2	2.5	3	
	Calculated Stiffness [N m/rad]			
Type A	80.3	75.5	85.7	80.5
Type B	74.9	90.2	90.7	85.3
Type C	191.5	N/A	255.9	223.7

8.4.2 Analysis

The calculated stiffnesses of the two hex bit sockets Type A and B did not differ by as much as was anticipated, despite their geometric differences. The experiment however shows that the Type C socket is stiffer than the other sockets which aligns with the hypothesis.

The fact that the two hex bit sockets resulted in the same stiffnesses could mean that some other factor is determining the stiffness between hammer and screw head. The socket can be seen as multiple torsional spring elements acting in series, the combined stiffness of which is given by Equation 13. According to this equation the element with the lowest stiffness will be limiting the combined stiffness.

The least stiff part of the socket Type A looks to be the slim shaft. The shaft is roughly 20 mm long with a diameter of 7.5 mm and a shear modulus of 80 GPa retrieved from GRANTA Edupack (Ansys, Inc, 2021) for a CrMo Steel. Using these values the stiffness of the shaft can be calculated according to equation 11 and 12 which results in a stiffness of 1240 N m/rad. This value is a lot greater then the one retrieved from the experiment for the Type A socket in table 2. This suggests that there is something else limiting the stiffness between the hammer and anvil.

The fact that the calculated stiffness for the Type C socket is almost twice of the Type A and B points towards the interface between socket and screw head being the determining factor for the stiffness. The impact of the stiffness will not be analysed further in this work.

8.4.3 Discussion

The results from the experiments are heavily influenced from the assumption that all of the rotational energy in the hammer is converted into twist in the spring between hammer and screw head. If there are losses in the impact, which should be expected, all of the angular velocity will not be converted to torque. This would lead to a smaller ω_{hammer} in equation 18 and therefore a higher stiffness in between the hammer and screw head.

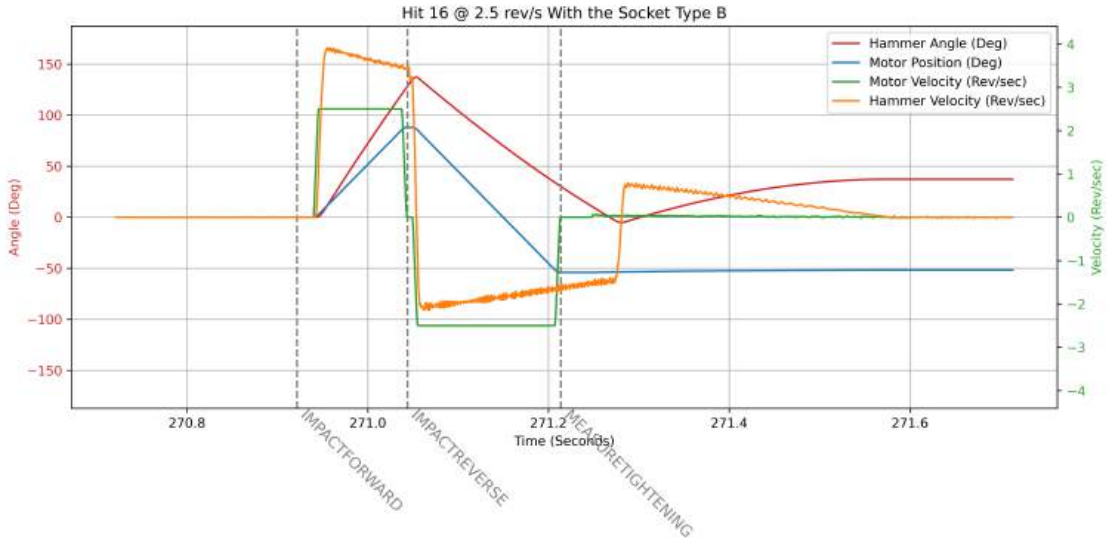

Figure 27: Plot of the 16th hit at 2.5 rev/s with socket type B.

Looking at the orange line in Figure 27 above it is clear that there are losses in the impact. The graph shows the 16th impact using socket type B. Despite there being no further tightening of the screw, the hammer loses almost 1.5 rev/S from the impact moment when comparing the impact speed to the recoil speed after the hit. This makes it hard to choose an angular velocity when calculating the stiffness, as there is likely losses both when the angular velocity is absorbed and when the energy is being released back into the hammer as recoil.

The fact that the stiffness has a major impact on the torque that the rig can apply to the screw makes it highly interesting to analyse it further. Future works could investigate the use of FEM-software to simulate the stiffness in the interface between socket and screw head. This would give greater understanding about the rig and the impact of using different sockets as well as the impact of tightening screws with worn out screw heads. The experiment could also be performed with a higher number of impact velocities and doing multiple tests with each socket and velocity. This would make it possible to investigate the character of the stiffness regarding whether or not it remains constant at different impact velocities.

9 Realisation

In this section the realisation of different estimation methods are covered. The result, analysis and discussion is contained for each realisation to improve legibility.

9.1 Preparing specimen

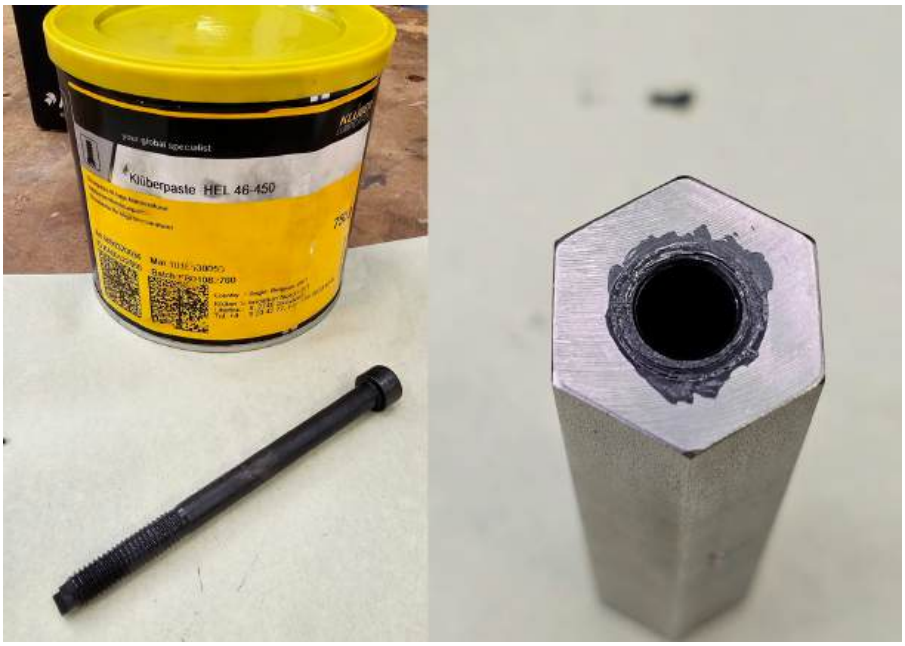

Figure 28: Left: The Klüberpaste can and screw with applied lubricant. Right: The flange with Klüberpaste applied to the surface where the screw and flange interact.

Before each realisation the specimen was prepared by cleaning both the thread in the flange and the screw with brake clean and blowing it dry using compressed air. This was done to remove any debris and residue from old lubricant within the thread. To achieve some control in regards of friction the lubricant ”Klüberpaste hel 46-450” is applied to the specimen(Klüber Lubrication, 2016). According to the product information it should result in a friction coefficient of $\mu = 0.1$. The lubricant is applied to both the thread of the screw and the under head surface on the flange, the application can be seen in Figure 28.

9.2 Realisation 1

This realisation combines multiple methods to estimate the preload of a specimen. First the initial screw length is recorded with the ultra sonic sensor after which, the rig is used to tighten the screw with one hit at varying velocities. This is followed by estimating the applied torque using the nutrunner with the re-tightening torque technique presented in Section 6.4.1. This sequence was repeated for three different motor speeds; 3, 4 and 5 rev/s and the results are shown in Table 3. In this realisation specimen type A was used.

9.2.1 Results

Table 3: Results from realisation 1. The tool or equation used to measure or calculate value is in parentheses next to the parameter name

Motor speed (Rig)	rev/s	3	4	5
Tighening angle (Rig)	°	15.3	19.6	31.7
Re-tightening torque (Nutrunner)	N m	4.19	6.72	7.58
Length before rig (Ultrasonic)	mm	104.75	104.75	104.73
Length after rig (Ultrasonic)	mm	104.88	104.96	105
Length diff after rig (Ultrasonic)	mm	0.13	0.21	0.27
Length after nutrunner (Ultrasonic)	mm	104.92	104.98	105.02
Rig-nutrunner length diff (Ultrasonic)	mm	0.04	0.02	0.02
Angle (Equation 7)	kN	4.9	6.3	10.2
Stretch (Equation 5)	kN	14.9	24.1	30.9
Torque (Equation 14)	kN	3.6	5.8	6.5

9.2.2 Analysis

Assuming the preload value derived from the measured angle is the most accurate it is relevant to look how much the other estimations deviate.

Table 4: Table of the deviation from the angle preload for the stretch and torque methods

Motor speed	rev/s	3	4	5
Deviation stretch	%	203.8	283.1	204.6
Deviation torque	%	-26.6	-8.1	-35.9

Table 4 shows how much the stretch and torque methods deviate from the angle estimation in percent. The Root Mean Square Percentage Error or RMSPE comes out to 233.5% for the stretch method and 26.2% for the torque method.

9.2.3 Discussion

It is clear that the preload calculated from the ultra sonic sensor is of by a large margin, this is analysed and discussed in the next realisation.

The retightening torque method shows promise with a RMSPE of 26.2%. Reasons for the deviation could both stem from human error when measuring and parameters such as the friction being wrong. If the friction in the joint is lower than the one assumed the estimations would show higher values, closer to the direct angle estimation. In realisation 3 a friction coefficient of 0.07 instead of 0.1 is tried resulting in better values. If 0.07 is used here the estimated preloads with the retightening torque method comes out to 4.79, 7.69 and 8.67. These values correspond to a RMSPE of 15.5%. The retightening torque technique also fully relies on the operator stopping at the same position each time to get the correct torque. Being off by just 3 degrees converts to a preload of about 1 kN.

9.3 Realisation 2

The following realisation is using a specially designed setup provided by Nils Dressler at Atlas Copco. Only one tightening was performed which is worth keeping in mind when reading the results of the test.

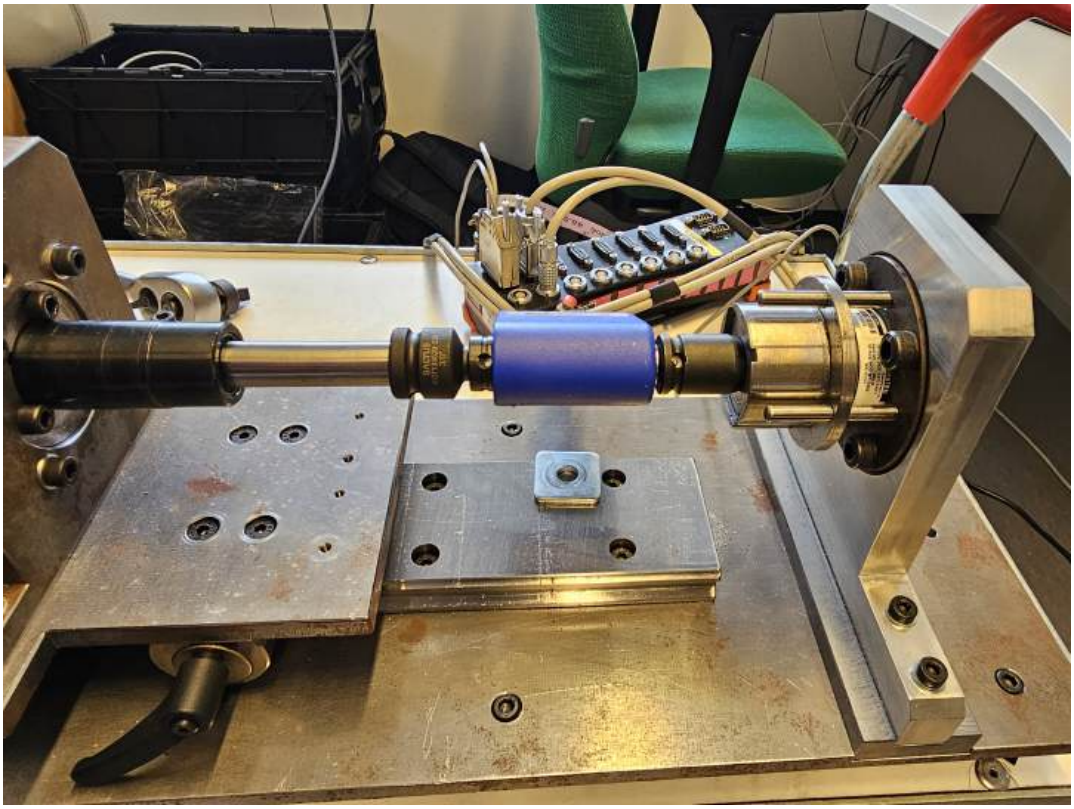

Figure 29: Nils Dressler's special tightening test rig.

The rig, seen in Figure 29 above, is equipped with a torque transducer and a load cell that outputs their measurements to a program called Dewesoft. This makes it possible to validate the ultrasonic sensors accuracy. During the test the length of the bolt was recorded with the ultrasonic sensor before and after tightening and after untightening it again.

9.3.1 Results

Table 5 displays all the measured parameters from the test.

Table 5: Results from test 2

Length before tightening	mm	74.39
Length after tightening	mm	75.04
Difference in length	mm	0.65
Length after untightening	mm	74.39
Peak Torque	Nm	22.73
Tightening angle	°	40.0
Measured preload	kN	6.62

9.3.2 Analysis

The stiffness of the specific bolt used in this test can be calculated using Equation 1 since it is completely threaded. The values for d_1 and d_2 for an M8 bolt are 6.647 mm and 7.188 mm respectively (IEI/Maskinkonstruktion, 2017). These values and Equation 2 gives an area of 37.582 mm^2. With this area, a clamped length of 61.42 mm and a Young's modulus of 210 GPa, Equation 1 gives a stiffness of 128.50 kN/mm. This stiffness together with the measured difference in length of 0.65 mm gives a preload of 83.52 kN using Equation 5.

9.3.3 Discussion

The calculated preload is much greater than the one measured by the load cell, which is most likely due to the measured stretch being too large. Both the fact that the measured length after untightening exactly corresponded with the one before and the relatively low measured preload show that no plastic deformation took place. According to Verein Deutscher Ingenieure (2003), an M8 8.8 bolt tightened to 90% of its elastic limit should be subject to a preload of around 19 kN. Calculating backwards with Equation 5 this gives an approximate stretch of 0.148 mm. The measured 0.65 mm is comparatively much greater than the expected value. When comparing the initial bolt measurement with a digital caliper the values lined up exactly. After tightening it was, however, impossible to validate using the caliper due to the setup of the rig.

The misleading result of the Ultrasonic measurement indicates that something is not correct with the equipment. This could be a calibration issue; the short time with the equipment at hand meant that there was no time to learn the ins and outs of it. The correct measurement of the untightened bolt points towards the speed of sound in the material without stress being correct. However, the decrease of the speed of sound in the material could be wrong or not taken into account.

When reading the documentation of the Panametrics-NDT 35 it was noted that there is no mention about screws, bolts, or doing the types of measurement that we are interested in. The equipment is stated to only be a thickness gauge for different materials. This could mean that the equipment does not take into account the difference stress in the material has on the speed of sound.

9.4 Realisation 3

In realisation 3 the newly acquired screw length measurement rig was used in combination with the tightening-untightening nutrunner method presented in Section subsection 6.5. The nearly unused specimen type B was used in this realisation. The goal of this test was to validate the relative accuracy and consistency between the two methods. The screw was tightened using the nutrunner to different torques starting with a target torque of 5 N m. After which the stretch of the screw was measured using the dial gauge. After measuring the stretch, the tightening-untightening method was used. These steps were then repeated three times for torques ranging from 5 to 10 N m with steps of 0.5 N m meaning 33 runs were made in total.

9.4.1 Results

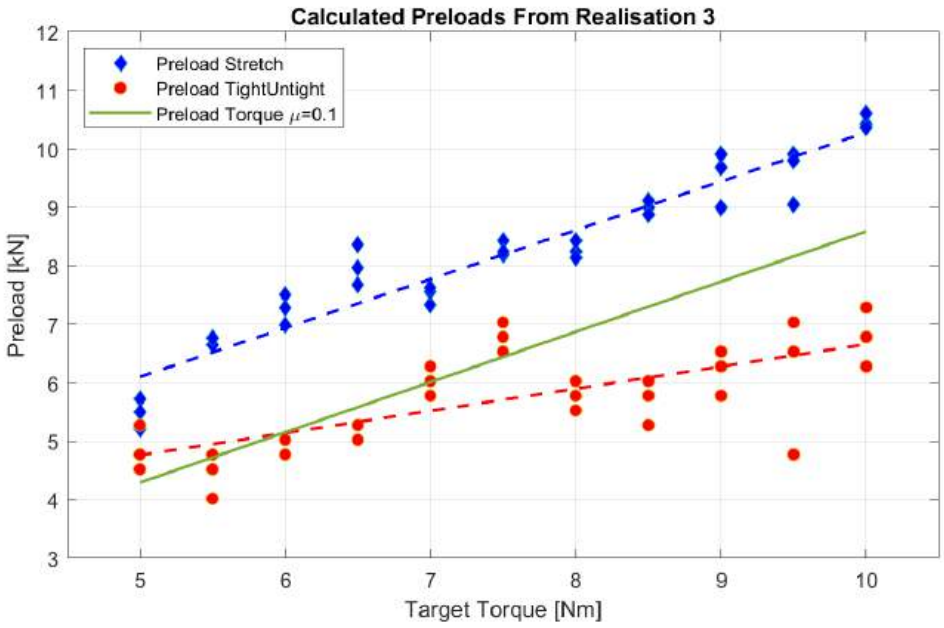

Figure 30: Scatter plot of the results from realisation 3 with line fits for each data set. The green line is the preload according to Equation 14 with $\mu_t = \mu_n = 0.1$.

The calculated preload using the dial gauge and the tightening-untightening methods are displayed in Figure 30. The values from the dial gauge and the tightening and untighten-

ing torque from the nutrunner, together with the calculations used to estimate the preload, can be found in Appendix C.2.

9.4.2 Analysis

The results in Figure 30 shows a discrepancy between the estimated preload from the measured stretch and the tightening-untightening torque. The second of which also shows a greater spread and more irregularities compared with the preload derived from stretch.

The result from the stretch estimation resulted in a higher preload than the one calculated using the target torque. However the slope of the fitted line matches the green line fairly close. Looking at the preload from the tightening-Untightening method the result shows a greater spread and a lower slope compared to the green line. It would be expected that the fitted line to the different estimated preloads should go towards zero when the target torque goes to zero. This however does not seem to be the case for the tightening-untightening torque whose line is really flat.

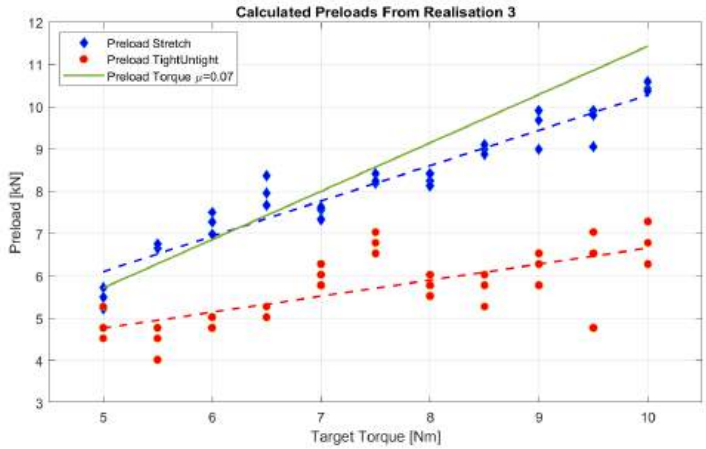

Figure 31: Scatter plot of the results from realisation 3 with line fits for each data set. The green line is the preload according to Equation 14 with $\mu_t = \mu_n = 0.07$.

The preload estimated by the target torque assumes that the friction coefficient achieved in the thread and under the head of the screw is constant and known, at each target torque. If the preload calculated by the stretch is correct that suggest that the friction coefficient used is incorrect. By changing the friction constant the green line can be adjusted to better fit with the preload calculated by the stretch of the bolt. In Figure 31 above the friction used to calculate the preload from the target torque is changed to 0.07. This makes the green line a closer fit to the preload calculated from the stretch of the screw.

Figure 32: The specimen after having been tightened and untightened 33 times. An area without any lubricant can be seen where the screw and flange interact.

When measuring the stretch of the bolt it was noted that the measurement did not follow the expected linear progression from the first target torques. This can also be seen in Figure 31, for the first 4 target torques the stretch preload seems to increases linearly. After that the increase in preload with each target torque decreases. After all of the measurements were performed the specimen was inspected. It was clear that the repeated tightening and untightening had pushed out most of the lubricant from under the head of the screw, this can be seen in Figure 32. To investigate the effect this has on the result, the last set of the experiment with a target torque of 10 N m was recreated with a re-lubricated screw. The results from this test compared to the original tightening can be seen in Table 6 below.

Table 6: Comparison between a the final tightening in the experiments and the re-lubricated screw. The calculated preload from the stretch of the screw

	Preload by stretch kN			
Target torque N m	1st	2nd	3rd	Average
10	10.4	10.4	10.6	10.5
10 (Re-lubricated)	12.3	10.8	11.4	11.5

Re-lubricating the screw increased the average preload over three tightenings with almost 10%. The first tightening with the re-lubricated screw resulted in an almost 20% increase in preload. This shows the variance that can be expected when tightening seemingly the same screw joint from time to time.

9.4.3 Discussion

The result from the tightening-untightening estimation method was not deemed satisfactory as it highly varies from the expected preload at each target torque. Implementing the

method comes with some difficulties regarding how to implement it using the nutrunner. The first problem comes from the fact that to measure the torque needed to turn the screw further, the screw needs to actually be turned further. This will increase the preload in the screw joint. The screw will then need to be loosened to find the untightening torque. It is not trivial how to achieve the untightening torque as the screw has been turned from its initial position.

The implementation on the nutrunner is made difficult by the stiffness between the screw head and the encoder on the tool. The encoder will read an increasing angle as the tool starts to apply torque to turn the screw. This angle that the encoder reads, without any physical turn of the screw, will be larger the larger the torque needed to tighten or untighten the screw is. This means that the program used can not be set to just turn the bolt a certain angle in each direction as that does not guarantee any rotation of the screw. This is then followed by the fact that there will be some static friction that is greater than the dynamic friction. This means that the so-called ”break away torque” will be the one recorded when tightening the screw further. When untightening the screw past its initial position, it will be the dynamic friction that is measured.

This method still shows promise and might be able to provide better result if the difficulties above can be circumvented. This could be of interest for future projects to find a better way of implementing the method on the nutrunner to get around the problems stated here.

Using the dial gauge and the new measurement rig made it possible to get a fairly high resolution for the elongation of the screw. The resolution of the dial gauge is half a micron; however, other factors make this resolution hard to achieve. When using the rig, although there is a rail for the specimen to be placed in, this does not fully constrict the specimen. There is some rotational play between the specimen and the measurement rig, moving the specimen through this play will sometimes alter the measured stretch by 8 microns and sometimes there would not be any perceived difference by moving the specimen.

In Figure 30 it is clear that there is a spread of the estimated preload at each target torque using the stretch. This could be attributed to the reasons stated above. It could also be an indication of the fact that the friction during tightening varies between each tightening. The bolt used together with the measurement rig is flattened at the threaded side to have a nice surface for the dial gauge to measure against. It is not known if this surface is orthogonal to the centre axis of the screw. If this is not the case, this could mean that the screw rotation can influence the measurement from the dial gauge.

9.5 Realisation 4

A special test run with the impact drive rig was performed in order to try the tighten-untighten torque technique with torque estimations just as explained in Section 6.6.2. This test used socked type A and specimen type A. In total four separate hits where made. First a tightening hit at a motor speed of 2.5 rev/s tightening the screw 10.7 degrees was made. Figure 33 below showcases the hit in full. The interesting parameter to pick out here is the recoil speed of the hammer which in this case was 0.69 rev/s or 4.34 rad/s.

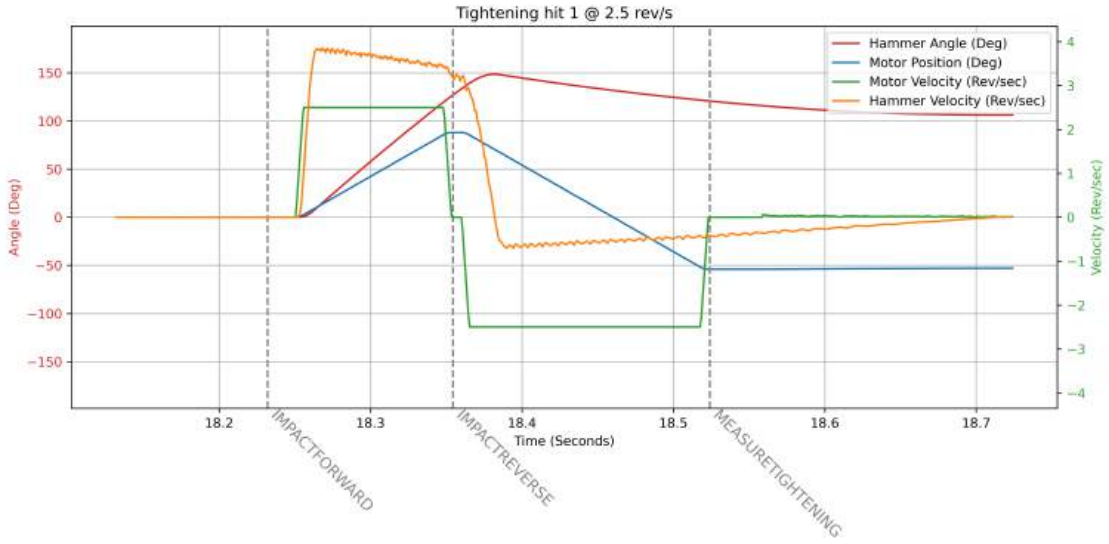

Figure 33: A tightening hit with a motor speed of 2.5 rev/s.

Using Equation 17 with a hammer inertia of 0.00284 kgm^2 and a torsional stiffness $K_{torsion}$ of 80.5 Nm/rad calculated in section 8.4.2 the approximated impact torque for this hit comes out to 2.07 Nm.

This was then followed by an untightening hit with a recoil speed of 2.51 rad/s giving a new impact torque of 1.20 Nm. This hit untightened the screw by 2.3 degrees. After this one more tightening hit increasing the angle 6.3 degrees and one untightening hit decreasing the angle 1.7 degrees again were made. These second hits gave tightening and untightening recoil speeds of 5.03 rad/s and 3.46 rad/s respectively. Converted to torque with Equation 17 gives 2.40 Nm for the tightening and 1.65 Nm for the untightening.

9.5.1 Results

Table 7: Results of Realisation 4

	Angle / °	Recoil speed / rad/s	Impact torque / N m
Tightening 1	10.7	4.31	2.07
Untightening 1	2.3	2.51	1.20
Tightening 2	6.3	5.03	2.40
Untightening 2	1.7	3.46	1.65

9.5.2 Analysis

Having calculated both the tightening and untightening impact torques it is now possible to use Equation 15 to get an estimation of the preload which comes out to 2.19 kN for the first two hits and another 1.89 kN added by the second hits.

To compare these values another estimation using the tightening angle and Equation 7 could be made. $K_J = 92.3$ MN/m is the joint stiffness as presented in section 3.1. The preload for the first hit using this method comes out to 3.43 kN and 2.02 kN for the second hit.

9.5.3 Discussion

The ”tightening-untightening torque” method using the hammer inertia resulted in values that are a bit lower than those calculated using the tightening angle from the rig. Using this method on the rig, the problems from the previous realisation on the nutrunner are still present. It is not trivial what angle should be compared to the estimation from the ”tightening-untightening torque” as the screw is turned by the untightening hit. The calculations also come with some uncertainties associated with the stiffness used to calculate the torque applied from the recoil velocity.

As the method shows promise for making an estimation, it would be of interest to further investigate it by conducting more experiments to acquire additional data that can be analysed. The method could be improved by starting off with a low impact speed when tightening and untightening the screw. The speed can then be increased until a change in angle can be detected.

9.6 Realisation 5

An implementation of the estimation by screw twist method, found in section 6.7, is explored in this realisation. Here, the difference in angle given by the Sick camera at the tip of the screw and the angle from the rig at the head is the twist angle of the screw. The realisation was conducted using specimen type A. Two tightenings for motor speeds of 3, 4, 5 and 6 rev/s were performed. Using the 341 Nm/rad torsional stiffness of the screw presented in section 3.1 and the difference in angle from the results converted to radians Equation 10 gives a torque Nm which gives a preload. The preload is also estimated directly from the rigs tightening angle to have something to compare with.

9.6.1 Results

Table 8: Results from realisation 5 with measured angles and calculated torques and preloads at different motor speeds

Motor speed rev/s	Cam angle °	Rig angle °	Delta angle °	Torque Nm	Twist preload kN	Angle preload kN
3	19.34	19.69	0.35	2.07	3.37	6.31
3	18.60	18.97	0.37	2.20	3.58	6.08
4	26.23	26.57	0.35	2.06	3.35	8.52
4	25.39	25.82	0.43	2.54	4.13	8.27
5	29.53	29.89	0.36	2.13	3.46	9.58
5	30.21	30.71	0.50	2.98	4.86	9.84
6	35.35	35.87	0.52	3.08	5.02	11.50
6	34.40	34.95	0.54	3.23	5.26	11.20

9.6.2 Analysis

The estimation made directly from the tightening angle can be used as a validation of the twist method.

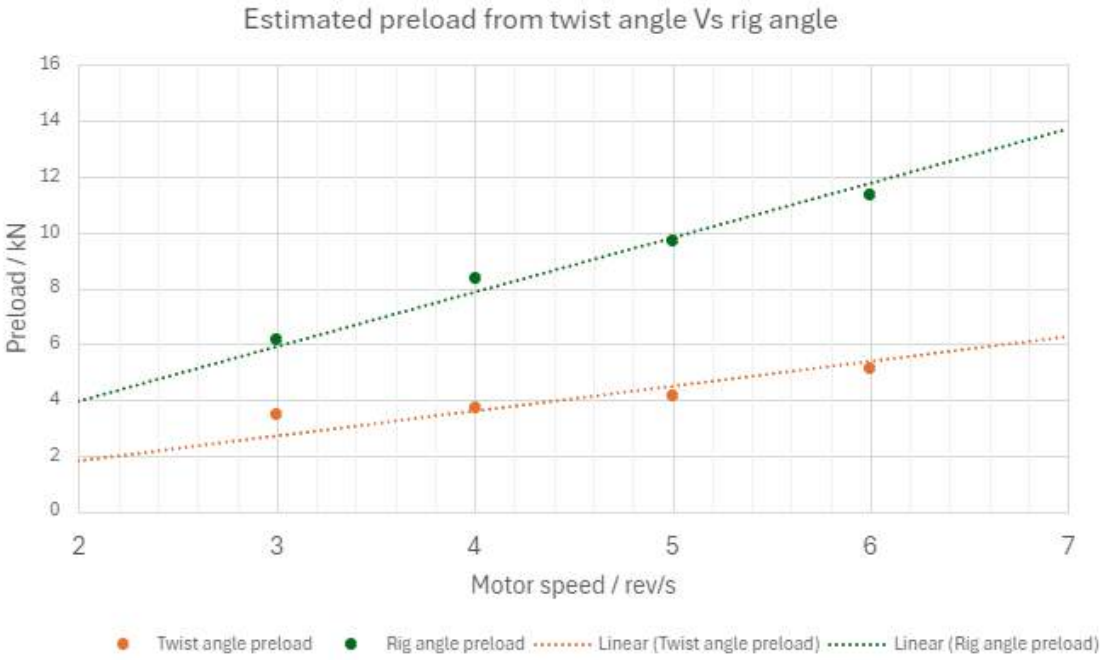

Figure 34: Comparison plot of the preload estimated from camera and rig angle.

In Figure 34 a line has been fitted for both the estimated preload from the twist angle and the rig angle. Each dot is the mean of the two values recorded at each motor speed calculated from the values in Table 8. The RMSPE for the twist estimation when compared against the rigs tightening angle estimation is 53%.

9.6.3 Discussion

The data in Table 8 shows that the camera consistently reads a lower tightening angle than the hammer encoder, as would be expected. The difference in angle between the screw head and the thread also increases with the increase in impact velocity, proving that the rig and the camera are capable of providing data for an analysis regarding the twist of the bolt.

Looking at the linear regressions, there seems to be a static offset error in the twist estimation. This could be because of the many unknown parameters in the equations. Friction is one of the unknown parameters assumed to be 0.1 according to the data sheet of the Klüberpaste. As seen in other realisations, it appears that the friction present in the joint is lower than 0.1. A lower friction would yield a higher calculated preload for the twist method, making the result closer to the one calculated from the tightening angle. The

torsional stiffness of the bolt also directly contributes to the result of the twist method, and changing it could yield a more favourable result.

The preload estimations made here rely on the accuracy of both angle measurements. An analysis of the camera's accuracy was made by the rig improvement group (Johansson and Kämpe, 2024). For more information regarding the accuracy of the camera, see their report. The accuracy of the angle measured from the rig has not been closely analysed but is assumed to be sufficient for most methods used in the project, including this one.

It is not known what happens to the twist of the bolt after the tightening. It could be that some of the twist generated while the bolt is turned goes away when the applied torque is removed. It is not trivial what torque is responsible for the twist of the screw. In the realisation, it is assumed that it is the thread friction and the bolt stretch component that contributes to the twist, as stated by Bickford (2008). If the thread is assumed to be held fixed by friction and the friction is lower in the head, the torque from the bolt twist should not be greater than the torque in the head of the bolt. This would have to be further investigated.

10 General Discussion

Multiple methods for estimating the preload have been presented and investigated in this report. The methods range from more straightforward ones based on torque, angle, and bolt stretch, to more intricate ones focused on the impact drive rig. These use the angular velocity of the hammer's recoil and the hammer inertia to make an estimate of the torque applied to the screw. This torque can then be used to estimate the preload. Some of the presented methods could not be realized due to time restraints.

The realisation of the different estimation methods should be considered as a ”proof of concept”. The purpose of this was to evaluate their practicality and find shortcomings that need to be taken into account when implementing the methods. When evaluating the results from the realisations, it is important to note that it is the spread of the results that is unwanted. If the estimation method results in a constant error, it would be possible to compensate for with an error coefficient. It should be noted that a constant error could also be the cause of assumptions made for the specific method.

The workings of a screw joint are commonly represented as multiple springs that are stretched and compressed. However, that representation is not the easiest to wrap one's head around. The screw length measurement rig made the preload achieved when tightening the screw more tangible than the other methods. Being able to turn the screw with a simple hand tool and directly measure an elongation of the screw is a great way of illustrating this concept.

Access to the equipment needed to realise methods was limited during the project. For the rig, this was primarily due to further development. In the later stages of the project, the motor was handed over to another project. This occurred at the same time as the length measurement rig was completed by the workshop. This meant that it was not possible to carry out all of the proposed methods together in one test. The ultrasonic equipment was only available during one afternoon leading to great restrictions regarding the tests that could be executed. Hopefully, future projects will be able to get more hands-on time with the equipment, resulting in more tests to evaluate and draw conclusions from.

10.1 Ethical Reflection

It is of great importance to take the ethical aspects into account when conducting any project. In this project, there will not be an end product meant for consumers. The main purpose was to continue development of the already existing impact drive rig, a rig meant for experiments and internal use at Linköping University.

The many parameters that affect a bolted joint and the preload achieved when tightening it lead to screw joints being oversized. This is done for the safety and integrity of the construction but comes at the cost of needing a greater number of screws or the use of larger screws. This, in turn, leads to unnecessary use of our planet's resources. Increasing the understanding about screw tightening and finding ways of making good estimations of the preload can also ensure safer constructions to mitigate the risk of failure.

It is possible that the rig will become relevant in future labs taking place in courses about screw connections. That would mean that students who have no prior knowledge about the equipment would be using it. With its strong motor providing the ability to spin at great speeds and generate large forces, it is of great importance that there is good documentation provided when handing over the rig. Both regarding how to use it, but primarily, how to use it in a safe manner.

In this report, multiple ways of determining the preload in a bolted joint are investigated and evaluated. The tests performed are only done to a short extent and the results are not validated, neither do they reflect the full story. They should therefore be seen as findings to build further knowledge and understanding upon and not as the truth to be used in practice.

11 Conclusions

In this section, the research questions will be answered, and recommendations for future work will also be presented.

RQ1: How can one estimate the preload in bolted fasteners?
There are many different tools and methods to estimate the preload, all stemming from a few equations derived from classical screw mechanics. The basic ones include measurement of turn angle, applied torque, and bolt stretch. Other, more intricate methods have also been explored, such as recording both tightening and untightening torques, knowing the recoil inertia of an impact drive system, and measuring the twist of the screw.

RQ2: What methods could be implemented on the current impact drive rig?
The methods that could be realized were estimation by tightening angle, tightening-untightening torque from recoil inertia, stretch measurement, and screw twist. Most of the methods showed promising results and should be investigated further.

RQ3: What accuracy can be achieved by the different estimation methods?
The accuracy of the methods implemented was not fully established due to time restraints. However, the result from using the newly designed length measurement rig proved good consistency and is recommended as a reference for validating other methods. For each realization, more data points are needed to draw real conclusions about accuracy.

11.1 Recommendations of Future Work

Here are some areas of this project that could be further investigated to gain further knowledge into the experimental rig and the general understanding about screws.

- A deeper analysis into the interface between the socket and screw head could be made using FEM. This could give a better understanding of what is determining the stiffness of the system and the effect of using the different types of screws.
- Finding a way of implementing the Tightening-untightening method to get around the problems with static and dynamic friction, as well as the stiffness of the nutrunner to get better results.
- Making further analyses to find out the reason behind the dip in velocity right before the impact between hammer and anvil. This would increase the understanding about the rig.
- Repeating all the realisations to get more data and then map out the spread and accuracy for each one.

References

Alfredsson, B. (2014). *Handbok och formelsamling i Hållfasthetslära*. Instutitionen för hållfasthetslära KTH.

Ansys, Inc (2021). GRANTA EduPack [Computer Software]. https://www.ansys.com/products/materials/granta-edupack. Version 21.1.2.

Appeltofft, N., Barisic, A., and Trossmark, R. (2023). *Simulation of an Impact Drive Mechanism for the Tightening and Untightening of Screw Connections for Research Purposes*. Bachelor's thesis, Linköpings Universitet.

Atlas Copco (2018). *Power focus 6000 - torque controllers: Atlas Copco*. https://www.atlascopco.com/en-us/itba/products/assembly-solutions/electric-assembly-systems/powerfocus-6000. Accessed: 12 April 2024.

Atlas Copco (2021). *8436616070 ETV STR61-70-13*. https://servaid.atlascopco.com/AssertWeb/en-US/AtlasCopco/Catalogue/4784. Accessed: 11 April 2024.

Augustini, J., Juell-Skielse, J., and Lidbäck, E. (2022). *Thightening and Unitghtening of Heavy-Duty Screw Connections*. Bachelor's thesis, Linköpings Universitet.

Bickford, J. H. (2008). *Introduction to the Design and Behavior of Bolted Joints, Non-Gasketed Joints*. CRC Press, Taylor & Francis Group.

B&R (2024a). *B&R Driver Product Page*. https://www.br-automation.com/en-us/products/8ei8x8hws100600-1/. Accessed: 2024-03-24.

B&R (2024b). *B&R Motor Product Page*. https://www.br-automation.com/en-us/products/8lsa46db060s300-3/. Accessed: 2024-03-24.

B&R (2024c). *B&R PLC Product Page*. https://www.br-automation.com/en-us/products/plc-systems/x20-system/x20-plc/x20cp1684/. Accessed: 2024-03-24.

B&R (2024d). *B&RAutomation Studio Product Page*. https://www.br-automation.com/en-us/products/software/automation-software/automation-studio/. Accessed: 2024-05-14.

Eriksson, C., Lindström, L., and Wärn, W. (2022). *The Effects of Artificial Ageing on Disassembly of Heavy-Duty Screw Connections - Design of a screw connection and corrosion method*. Bachelor's thesis, Linköpings Universitet.

Git (2024). *What is git?* https://git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F. Accessed: 2024-05-14.

Gitlab (2024). *About Gitlab*. https://about.gitlab.com/. Accessed: 2024-05-14.

Gustavsson, L. and Erlandsson, V. (2023). *Impulse Drive for Thightening and Unitghtening of Heavy-Duty Screw Joints*.

IEI/Maskinkonstruktion (2017). *Formelsamling Maskinelement*. LITH.

Johansson, A. and Kämpe, S. (2024). *Improving on Impulse Drive - Further development on an experimental impact driver and expanding system possibilities*. Bachelor's thesis, Linköpings Universitet.

Jupyter (2024). *Jupyter*. https://jupyter.org/. Accessed: 2024-05-14.

Klüber Lubrication (2016). Klüberpaste hel 46-450, prod. 0890232.

Laoyan, S. (2024). What Is Agile Methodology? (A Beginner's Guide) [2024]. https://asana.com/sv/resources/agile-methodology. [Accessed 09-05-2024].

Mathworks (2024a). *Simulink product page*. https://se.mathworks.com/products/simulink.html. Accessed: 2024-05-14.

Mathworks (2024b). *What is Matlab?* https://se.mathworks.com/discovery/what-is-matlab.html. Accessed: 2024-05-14.

Mitutoyo (2022). *Digimatic Mätur ID-C*. https://shop.mitutoyo.se/web/mitutoyo/sv_SE/mitutoyo/1300442458734/Digimatic%20M%C3%A4tur%20ID-C/$catalogue/mitutoyoData/PR/543-700B/index.xhtml. Accessed: 7 May 2024.

Olsson, K.-O. (2006). *Maskinelement*. Liber AB.

Python (2024). *What is Python? Executive Summary*. https://www.python.org/doc/essays/blurb/. Accessed: 2024-05-14.

RLS (2023). *LM10 Incremental Magnetic Encoder*. https://www.rls.si/eng/lm10-linear-and-rotary-magnetic-encoder-system#downloads|data-sheet. Accessed: 2024-03-24.

Shoberg, R. S. (2000). Engineering fundamentals of threaded fastener design and analysis. i. *Fastening*, 6(2):26–29.

SICK Sensor Intelligence (2024). *InspectorP61x Data sheet*. https://www.sick.com/us/en/catalog/products/machine-vision-and-identification/machine-vision/inspectorp61x/v2d611p-mmsbi4/p/p672485?tab=detail#technical-details. Accessed: 2024-03-24.

Tall, A., Hellström, A., and Alkestrand, J. (2022). *Heavy Duty Screw Connections – Testing Rig- A design and development project*. Bachelor's thesis, Linköpings Universitet.

Verein Deutscher Ingenieure (2003). *VDI-2230 Systematic calculation of high duty bolted joints. Joints with one cylindrical bolt*.

Appendices

Appendix A PLC Code

A.1 Main Program

```
PROGRAM _CYCLIC
    // Counting all program cycles
    cycle_counter := cycle_counter + 1;

    // sensor scaling
    hammer_sincos_angle_deg := -DINT_TO_LREAL(hammer_sincos_val) / 65536. *
        hammer_pole_len
    /(hammer_diameter * brmPI) * 360 - hammer_sincos_angle_deg0;
    motor_angle_deg := MpAxisBasic_0.Position * 360.0;
    MpAxisBasic_Position := MpAxisBasic_0.Position;
    MpAxisBasic_Velocity := MpAxisBasic_0.Velocity;

    // Stop button in mappView, Disable the motor. Resetted by 'Start motor' from View.
    IF visStop THEN
        MpAxisBasic_0.Enable := FALSE;
        cmdPower := FALSE;
        cmdJogTorqueEnable := FALSE;
        cmdJogTorqueStart := FALSE;
        cmdJogTorqueStop := TRUE;
        cmdMoveMotorBasic := FALSE;
        cmdJogBasicPositive := FALSE;
        cmdJogBasicNegative := FALSE;
        sStep := enINIT;
        opStepTightening := enOPTINIT;
        opCalibration := enCALINIT;
        MpAxisBasic_0.JogNegative := FALSE;
        MpAxisBasic_0.JogPositive := FALSE;
    END_IF

    // Catch errors in MpAxisasic
    IF MpAxisBasic_0.Error THEN
        visErrorCode := MpAxisBasic_0.StatusID;
        sStep := enERROR;
        opStepTightening := enOPTINIT;
    END_IF

    // Catch errors in Mc_BR_TorqueControl
    IF MC_BR_TorqueControl_0.Error THEN
        visErrorCode := MC_BR_TorqueControl_0.ErrorID;
        sStep := enERROR;
        opStepTightening := enOPTINIT;
    END_IF

    //MAIN STATEMACHINE
    CASE sStep OF
        enINIT:  //=================================================
            AxisParameters_Current := AxisParameters_ViewJogging;
            opStepTightening := enOPTINIT;
            opCalibration        := enCALINIT;
            IF visStartMotor THEN (*do nothing; sequence is started manually*)
                sStep:= enSTART;
            END_IF
```

```
enSTART: //=================================================
    MpAxisBasic_0.Enable := TRUE;

    (*is axis ready to be powered on?*)
    IF (MpAxisBasic_0.Active = TRUE) AND (MpAxisBasic_0.Info.ReadyToPowerOn =
        TRUE) THEN
        sStep := enPOWER_ON;
    END_IF

enPOWER_ON: //===============================================
    cmdPower := TRUE;

    (*axis powered on?*)
    IF MpAxisBasic_0.PowerOn = TRUE THEN
        sStep := enHOME;
    END_IF

enHOME: //===================================================
//      MpAxisHomingType_0.Mode := mcHOMING_DIRECT;
    AxisParameters_Current := AxisParameters_Calibration;

    cmdHome := TRUE;
    hammer_sincos_angle_deg0 := -DINT_TO_LREAL(hammer_sincos_val)
    / 65536. * hammer_pole_len /(hammer_diameter * brmPI) * 360;
    (*axis homed?*)
    IF (MpAxisBasic_0.IsHomed = TRUE) THEN
        sStep := enOPERATION;
        cmdHome := FALSE;
    END_IF

enOPERATION: //==============================================
    // joggimg
    AxisParameters_Current := AxisParameters_ViewJogging;

    cmdJogBasicPositive := visJogPos AND NOT visJogNeg;
    cmdJogBasicNegative := visJogNeg AND NOT visJogPos;
    paramLimitLoad := TRUE;

    IF visStartTightening THEN
        sStep := enTIGHTEN;
    ELSIF visStartUntightening THEN
        sStep := enUNTIGHTEN;
    ELSIF visStartCalibration THEN
        sStep := enCALIBRATE;
    END_IF

enTIGHTEN: //================================================
    parJogMaxTorque    := 0.6; // [Nm]
    parJogMaxVelocity := 1; // [rev/s]
    parJogMinVelocity := -0.5; // [rev/s]
    // Calls Tightening ACTION in Tightening.st
    Tightening;

enUNTIGHTEN: //==============================================
    parJogMaxTorque    := -0.6; // [Nm]
    parJogMaxVelocity := 0.5; // [rev/s]
    parJogMinVelocity := -1; // [rev/s]
    // Calls Untightening ACTION in Untightening.st
    Untightening;
```

```
enCALIBRATE: //=====================================================
    Calibration;

enERROR: //=======================================================
    (*power off*)
    cmdPower              := FALSE;
    (*reset all commands*)
    cmdMoveVelocity := FALSE;
    cmdMoveAdditive := FALSE;
    cmdUpdate             := FALSE;
    cmdHome          := FALSE;
    cmdJogBasicNegative := FALSE;
    cmdJogBasicPositive := FALSE;
    cmdJogTorqueEnable := FALSE;
    cmdMoveMotorBasic := FALSE;
    cmdMoveVelocity    := FALSE;
    MpAxisBasic_0.MpLink                  := ADR(gAxis_1);
    MC_BR_TorqueControl_0.Axis  := ADR(gAxis_1);
    IF visStartMotor THEN
        cmdErrorReset := TRUE;
    END_IF

    IF MpAxisBasic_0.Error = FALSE THEN;
        sStep := enSTART;
        cmdErrorReset := FALSE;
    END_IF
END_CASE

// Assign visOperationState as string value of current state
CASE sStep OF
        enINIT:
                visMainState    := "INIT";
                MainStateInt    := 1;
        enSTART:
                visMainState    := "START";
                MainStateInt    := 2;
        enPOWER_ON:
                visMainState    := "POWER_ON";
                MainStateInt    := 3;
        enHOME:
                visMainState    := "HOME";
                MainStateInt    := 4;
        enOPERATION:
                visMainState    := "OPERATION";
                MainStateInt    := 5;
        enTIGHTEN:
                visMainState    := "TIGHTENING";
                MainStateInt    := 6;
        enUNTIGHTEN:
                visMainState    := "UNTIGHTENING";
                MainStateInt    := 7;
        enCALIBRATE:
                visMainState    := "CALIBRATING";
                MainStateInt    := 8;
        enERROR:
                visMainState    := "ERROR";
                MainStateInt    := 9;
END_CASE

// Assign visOperationState as string value of current state
CASE opStepTightening OF
```

```
        enOPTINIT:
                OperationStateInt       := 1;
        enHOMEHAMMER:
                OperationStateInt       := 2;
        enHAMMERREPOS1:
                OperationStateInt       := 3;
        enHAMMERREPOS2:
                OperationStateInt       := 4;
        enMOTORREPOS:
                OperationStateInt       := 5;
        enIMPACTFORWARD:
                OperationStateInt       := 6;
        enIMPACTREVERSE:
                OperationStateInt       := 7;
        enMEASURETIGHTENING:
                OperationStateInt       := 8;
        enISFINISHED:
                OperationStateInt       := 9;
        enRESET:
                OperationStateInt       := 10;
END_CASE


// Basic motor control
MpAxisBasic_0.MpLink            := ADR(gAxis_1);
MpAxisBasic_0.Parameters        := ADR(AxisParameters_Current);
MpAxisBasic_0.Power                     := cmdPower;
MpAxisBasic_0.Home                      := cmdHome;
MpAxisBasic_0.ErrorReset    := cmdErrorReset;
MpAxisBasic_0.MoveAdditive      := cmdMoveMotorBasic;
MpAxisBasic_0.MoveAbsolute      := cmdMoveMotorAbs;
MpAxisBasic_0.LimitLoad     := paramLimitLoad;
MpAxisBasic_0.JogPositive   := cmdJogBasicPositive;
MpAxisBasic_0.JogNegative   := cmdJogBasicNegative;
MpAxisBasic_0.Update            := TRUE;
MpAxisBasic_0();

cmdMoveMotorBasic                       := FALSE;
cmdJogBasicPositive                     := FALSE;
cmdJogBasicNegative                     := FALSE;

MC_BR_TorqueControl_0.Axis                                  := ADR(gAxis_1);
MC_BR_TorqueControl_0.Enable                        := cmdJogTorqueEnable;
MC_BR_TorqueControl_0.Torque                        := parJogMaxTorque;
MC_BR_TorqueControl_0.TorqueRamp                    := 0.1;
MC_BR_TorqueControl_0.MaximumVelocity   := parJogMaxVelocity;
MC_BR_TorqueControl_0.MinimumVelocity   := parJogMinVelocity;
MC_BR_TorqueControl_0.Acceleration              := parJogTorqueMaxAcceleration;
MC_BR_TorqueControl_0.Start                         := cmdJogTorqueStart;
MC_BR_TorqueControl_0.Stop                                  := cmdJogTorqueStop;
MC_BR_TorqueControl_0.AdvancedParameters.CorrectVelocityLimits := TRUE;
MC_BR_TorqueControl_0();

// visualisation, reset buttons
visStartMotor           := FALSE;
visStop             := FALSE;
visStartTightening      := FALSE;
visStartUntightening := FALSE;
visStartCalibration  := FALSE;
visNextState            := FALSE;
```

```
        MC_ReadActualTorque_0.Axis := ADR(gAxis_1);
        MC_ReadActualTorque_0.Enable := TRUE;
        MC_ReadActualTorque_0();
        // Copy output variables
        write_output;

END_PROGRAM
```

A.2 Untightening Program

```
ACTION Untightening:
CASE opStepTightening OF

enOPTINIT: // ------------------------------------------------------
    visOperationState    := "OPTINIT";
    // Zero the impact parameters, to start a new measurement
    untightening_impact_counter := 0;
    untightening_angle_current_deg := 0;
    untightening_angle_diff_deg := 0;
    paramLimitLoad := FALSE;
    opStepTightening:= enHOMEHAMMER; //Initial state of positioning


enHOMEHAMMER://  ------------------------------------------------------
    visOperationState    := "HOMEHAMMER";
    // Start MPtorque jog
    // Jog torque-controlled until motor-hammer contact and hammer-anvil contact
    AxisParameters_Current  := AxisParameters_RePositioning;

    // Enable MPtorque
    cmdJogTorqueEnable  := TRUE;
    IF MC_BR_TorqueControl_0.Ready THEN
        cmdJogTorqueStart       := TRUE;
        cmdJogTorqueStop        := FALSE;
    END_IF

    // IF (Velocity within Min & Max limits AND Torque Limit is reached) for X cycles in
        a row
    // ==> Hammer homed successfully
    IF (NOT MC_BR_TorqueControl_0.AxisLimitActive) AND MC_BR_TorqueControl_0.InTorque
        THEN
        counter := counter + 1;
    ELSE
        counter := 0;
    END_IF

    IF counter >= 50 OR visNextState THEN // visNextState can be used to manually go
        through states
        cmdJogTorqueStart               := FALSE;
        cmdJogTorqueStop                := TRUE;
        cmdJogTorqueEnable              := FALSE;
        hammer_offset := hammer_sincos_angle_deg;
        motorPositionOffsetRev  := MpAxisBasic_0.Position;
        // Saves the motor position as an offset for data analysis purposes
        counter := 0;
        opStepTightening := enHAMMERREPOS1;
    END_IF


enHAMMERREPOS1: // ------------------------------------------------------
    visOperationState    := "HAMMERREPOS1";
    // Rotate motor cw to reposition hammer
    AxisParameters_RePositioning.Distance := GAP_MOTOR_TO_HAMMER_REV;
    AxisParameters_Current := AxisParameters_RePositioning;
    cmdMoveMotorBasic := TRUE;

    IF MpAxisBasic_0.MoveDone THEN
        opStepTightening := enHAMMERREPOS2;
        cmdMoveMotorBasic := FALSE;
```

```
    END_IF

enHAMMERREPOS2: // ------------------------------------------------------

    visOperationState    := "HAMMERREPOS2";
    AxisParameters_RePositioning.Distance := GAP_HAMMER_TO_ANVIL_REV;
    AxisParameters_Current := AxisParameters_RePositioning;
    cmdMoveMotorBasic := TRUE;

    IF MpAxisBasic_0.MoveDone THEN
        opStepTightening := enMOTORREPOS;
        cmdMoveMotorBasic := FALSE;
    END_IF

            (*
            // Next State by hand, testing
            IF visNextState THEN
                visNextState := FALSE;
                opStepTightening         := enMOTORREPOS;
            END_IF *)

enMOTORREPOS: // ------------------------------------------------------
    visOperationState    := "MOTORREPOS";
    // rotate motor cw to reposition motor
    AxisParameters_RePositioning.Distance := -(GAP_MOTOR_TO_HAMMER_REV +
        parMotorReposMarginRev);
    AxisParameters_Current := AxisParameters_RePositioning;
    cmdMoveMotorBasic := TRUE;

    //if movedone then
    // hammer-motor <= margin
    IF MpAxisBasic_0.MoveDone THEN
      ABS(hammer_sincos_angle_deg/360 - motorPositionMinusOffsetRev) >=
          motor_hammer_margin THEN
        visInfoText := 'Motor and hammer not in contact. Move hammer. Press confirm when
            done.';
        IF visHammerMotorInPosCorrect THEN
            visInfoText := '';  // empty string to view when position is fixed
            cmdMoveMotorBasic := FALSE;
            visHammerMotorInPosCorrect := FALSE; // Reset the 'button pressed' state
            opStepTightening := enIMPACTFORWARD;
        END_IF
    END_IF

enIMPACTFORWARD: // ------------------------------------------------------
    visOperationState    := "IMPACTFORWARD";
    // Accelerate motor and hammer
    // Assign high speed parameters and movement distance
    AxisParameters_Impact.Distance       := -(parImpactStrikingDistanceRev -
        parMotorReposMarginRev);
    AxisParameters_Impact.Velocity       := visUntighteningImpactVMax;
    AxisParameters_Current                      := AxisParameters_Impact;

    cmdMoveMotorBasic := TRUE;

    IF MpAxisBasic_0.MoveDone THEN
        opStepTightening := enIMPACTREVERSE;
        cmdMoveMotorBasic := FALSE;
    END_IF

    // Look into parameter updates (Lowering the acceleration for reversing)
```

```
enIMPACTREVERSE: // ------------------------------------------------------
    visOperationState    := "IMPACTREVERSE";
    // Reverse motor to avoid hammer recoiling into th motor driving pin.

    // Assign high speed parameters and movement distance
    AxisParameters_Impact.Distance       := parImpactStrikingDistanceRev +
        parImpactRecoilMarginRev;
    AxisParameters_Current                      := AxisParameters_Impact;
    cmdMoveMotorBasic := TRUE;

    IF MpAxisBasic_0.MoveDone THEN
        opStepTightening := enMEASURETIGHTENING;
        cmdMoveMotorBasic := FALSE;
    END_IF


enMEASURETIGHTENING:// ------------------------------------------------------
    visOperationState    := "MEASURETIGHTENING";
    // Both homing the hammer after impact, and measuring tightening angle
    // Jog torque-controlled CW until motor-hammer contact and hammer-anvil contact
    AxisParameters_Current := AxisParameters_RePositioning;
    cmdJogTorqueEnable  := TRUE;
    IF MC_BR_TorqueControl_0.Ready THEN
        cmdJogTorqueStart       := TRUE;
        cmdJogTorqueStop        := FALSE;
    END_IF


    // IF-condition checks if motor has moved too far,
    // ==> RESET to INIT state.
    // Motor position in rev

    (* FIXME:
    IF MpAxisBasic_0.Position  > 30.0/360 + parImpactRecoilMarginRev +
        GAP_HAMMER_TO_ANVIL_REV THEN
    cmdJogTorqueStart    := FALSE;
    cmdJogTorqueStop     := TRUE;

    opStepTightening     := enOPTINIT;
    sStep                               := enINIT;
    //Something was incorrect, reset to initial state enINIT
    END_IF
    *)

    // IF (Velocity within Min & Max limits AND Torque Limit is reached) for X cycles in
        a row
    // ==> Motor-hammer-anvil in contact
    // record tightening angle and move to next state
    IF (NOT MC_BR_TorqueControl_0.AxisLimitActive) AND MC_BR_TorqueControl_0.InTorque
        THEN
        counter := counter + 1;
    ELSE
        counter := 0;
        // If we can move an entire rev the untightening is finished
        IF (ABS(hammer_sincos_angle_deg - hammer_offset) > 360)
        OR (impact_counter >= visUntImpactAmountLimit)THEN
            cmdJogTorqueStart    := FALSE;
            cmdJogTorqueStop     := TRUE;
            cmdJogTorqueEnable   := FALSE;
            opStepTightening     := enISFINISHED;
```

```
        END_IF
    END_IF

    IF counter >= 50 OR visNextState THEN  // visNextState can be used to manually go
        through states
        cmdJogTorqueStart       := FALSE;
        cmdJogTorqueStop        := TRUE;
        cmdJogTorqueEnable      := FALSE;
        untightening_impact_counter := untightening_impact_counter + 1;
        //compare angle to previous untightening angle (before overwriting it)
        untightening_angle_diff_deg := (hammer_sincos_angle_deg -hammer_offset) -
        untightening_angle_current_deg;
        // save new untightening angle
        untightening_angle_current_deg := hammer_sincos_angle_deg - hammer_offset;
        // If we are not in free turn torque we need another hit
        opStepTightening        := enHAMMERREPOS1;
        counter := 0;

    END_IF


enISFINISHED: // ------------------------------------------------------
    visOperationState   := "ISFINISHED";
    // Record final untightening angle by torque jogging the screw back to snug
    parJogMaxTorque   := 0.6; // [Nm]
    parJogMaxVelocity := 1; // [rev/s]
    parJogMinVelocity := -0.5; // [rev/s]

    AxisParameters_Current := AxisParameters_RePositioning;
    cmdJogTorqueEnable  := TRUE;
    IF MC_BR_TorqueControl_0.Ready THEN
        cmdJogTorqueStart       := TRUE;
        cmdJogTorqueStop        := FALSE;
    END_IF

    IF (NOT MC_BR_TorqueControl_0.AxisLimitActive) AND MC_BR_TorqueControl_0.InTorque
        THEN
        counter := counter + 1;
    ELSE
        counter := 0;
    END_IF

    IF counter >= 50 OR visNextState THEN  // visNextState can be used to manually go
        through states
        cmdJogTorqueStart       := FALSE;
        cmdJogTorqueStop        := TRUE;
        cmdJogTorqueEnable      := FALSE;
        untightening_impact_counter := untightening_impact_counter + 1;
        // Compare angle to previous untightening angle (before overwriting it)
        untightening_angle_diff_deg := (hammer_sincos_angle_deg - hammer_offset) -
        untightening_angle_current_deg;
        // Save new untightening angle
        untightening_angle_current_deg := (hammer_sincos_angle_deg - hammer_offset);
        // Final angle recorded -> Reset
        opStepTightening        := enRESET;
        counter := 0;
    END_IF


enRESET: // ------------------------------------------------------
    visOperationState   := "RESET";
```

```
    // Move back the motor ccw so that the motor does not press against hammer and anvil
        .
    // rotate motor ccw to release contact between the components
    AxisParameters_RePositioning.Distance := - GAP_HAMMER_TO_ANVIL_REV /2;
    AxisParameters_Current := AxisParameters_RePositioning;
    cmdMoveMotorBasic := TRUE;

    IF MpAxisBasic_0.MoveDone THEN
        opStepTightening  := enOPTINIT;
        sStep                    := enOPERATION;
        cmdMoveMotorBasic := FALSE;
    END_IF
END_CASE
END_ACTION
```

A.3 Write Output Program

```
ACTION write_output:
        //1
        Output_0.cycle_counter_out := UDINT_TO_LREAL(cycle_counter);
        //2
        Output_0.MainStateInt_out := UINT_TO_LREAL(MainStateInt);
        //3
        Output_0.OperationStateInt_out := UINT_TO_LREAL(OperationStateInt);
        //4
        Output_0.hammer_angle_deg_out := hammer_sincos_angle_deg;
        //5
        Output_0.MpAxisBasic_Position_out := MpAxisBasic_Position;
        //6
        Output_0.MpAxisBasic_Velocity_out := MpAxisBasic_Velocity;
        //7
        Output_0.untightening_angle_deg_out := untightening_angle_current_deg;
        //8
        Output_0.untightening_impact_counter_out := UINT_TO_LREAL(
            untightening_impact_counter);
        //9
        Output_0.tightening_angle_deg_out := tightening_angle_current_deg;
        //10
        Output_0.impact_counter_out := UINT_TO_LREAL(impact_counter);
        //11
        Output_0.cam_angle_out := REAL_TO_LREAL(cam_angle);
        // 12
        Output_0.motor_torque_out := REAL_TO_LREAL(MC_ReadActualTorque_0.Torque);
END_ACTION
```

Appendix B Python Code

B.1 Preprocessing.py

```
import os
import sys
import h5py
import pandas as pd
import numpy as np
from numpy import r_

### Mapping state ints to strings
def state_int_to_string(state_int, type):
    """
    Maps state integer values from simulink to strings.
    """
    # Dictionaries for mapping integers to strings
    main_strings = {
        1: "INIT",
        2: "START",
        3: "POWER_ON",
        4: "HOME",
        5: "OPERATION",
        6: "TIGHTENING",
        7: "UNTIGHTENING",
        8: "CALIBRATING",
        9: "ERROR"
    }
    operation_strings = {
        1: "OPTINIT",
        2: "HOMEHAMMER",
        3: "HAMMERREPOS1",
        4: "HAMMERREPOS2",
        5: "MOTORREPOS",
        6: "IMPACTFORWARD",
        7: "IMPACTREVERSE",
        8: "MEASURETIGHTENING",
        9: "ISFINISHED",
        10: "RESET"
    }

    # Selecting the appropriate dictionary based on the type
    if type == "main":
        return main_strings.get(state_int, "Unknown State")
    elif type == "operation":
        return operation_strings.get(state_int, "Unknown State")
    else:
        return "Invalid Type"
```

```
def flatten_hdf5_dataset(dataset):
    # Read the dataset values (which are object references)
    object_references = dataset[0]

    # Create an empty dictionary to store the values
    values_dict = {}

    # Fill the dictionary with values referred to by the object
       references
    for index, ref in enumerate(object_references):
        # Get the name of the referred dataset
        dataset_name = f'dataset_{index}'  # Creating a generic name

        # Follow the reference and get the actual values
        referred_values = dataset.file[ref][()]

        # Store the values in the dictionary
        values_dict[dataset_name] = referred_values.flatten()

    return values_dict

### Main preprocessing function
def preprocess_data(file_path):
    with h5py.File(file_path) as f:
        settings_dict = {}
        data = {}

        impulse_data_group = f['impulsdata']

        if 'settings' in f:
            settings_dataset = f['settings']
            settings_dict = flatten_hdf5_dataset(settings_dataset)

        # Iterate over the variables in the group and store them in
           the dictionary
        for var_name in impulse_data_group.keys():
            # Reshape the variable to be 1D
            data[var_name] = impulse_data_group[var_name][:].flatten()

    # Close the HDF5 file
    f.close()

    # Convert dictionary to DataFrame
    experiment_data = pd.DataFrame(data)
    settings = pd.DataFrame(settings_dict)
    # Adding a time column to the DataFrame
    cycleTimeSec = 0.0008
```

```
experiment_data['timeSeconds'] = (experiment_data['CycleCounter']-
    experiment_data.CycleCounter.iloc[0]) * cycleTimeSec

# Converting motorPosition from Rev to deg, and subtracting offset
    from homing
experiment_data['motorPositionDeg'] = (experiment_data['
    MpAxisBasicPosition']) * 360.0

# Adding columns for the angular gap between the motor and hammer,
    and the hammer and anvil
GAP_MOTOR_TO_HAMMER_DEG = 311.98    # [deg] angular gap where the
    driving pins from the motor and hammer can rotate between ends
    of contact
GAP_HAMMER_TO_ANVIL_DEG = 138.3      # [deg] angular gap where
    hammer and anvil can rotate between ends of contact

#experiment_data['motorPositionPlusGapDeg'] = experiment_data['
    motorPositionDeg'] + GAP_MOTOR_TO_HAMMER_DEG
#experiment_data['tighteningAngleMinusGapDeg'] = experiment_data['
    TighteningAngleCurrentDeg'] - GAP_HAMMER_TO_ANVIL_DEG

# Create a new row by duplicating the last row
new_row = experiment_data.iloc[-1].copy()

# Assign the new row index (145739)
num_indices_column = len(experiment_data['CycleCounter'])
new_row.name = num_indices_column+1

# Add the new row to the dataframe
experiment_data.loc[num_indices_column+1] = new_row

# Applying the state_int_to_string function to the mainStateInt
    and operationStateInt columns
experiment_data['mainState'] = experiment_data['MainStateInt'].
    apply(state_int_to_string, args=("main",))
experiment_data['operationState'] = experiment_data['
    OperationStateInt'].apply(state_int_to_string, args=("operation
    ",))

experiment_data['state_change'] = experiment_data['
    OperationStateInt'].diff().fillna(0) != 0

return (experiment_data, settings)
```

B.2 Camera plot

```
angle_at_measure_hammer = experiment_data[experiment_data['
    operationState'] == 'ISFINISHED']['TighteningAngleCurrentDeg'].iloc
    [0]

filtered_data = experiment_data[(experiment_data['timeSeconds'] >=
    impact_forward_time-25) & (experiment_data['timeSeconds'] <=
    impact_forward_time+25)]
mean_cam_angle_before = experiment_data[(experiment_data['timeSeconds'
    ] >= impact_forward_time-25) & (experiment_data['timeSeconds'] <=
    impact_forward_time-5.5)]['CamAngle'].mean()
mean_cam_angle_after = experiment_data[(experiment_data['timeSeconds']
     >= impact_forward_time+1) & (experiment_data['timeSeconds'] <=
    impact_forward_time+25)]['CamAngle'].mean()
print(mean_cam_angle_before)
print(mean_cam_angle_after)
cam_tightening_angle = mean_cam_angle_after-mean_cam_angle_before
print("Tightening␣cam␣angle:", cam_tightening_angle)
print("Tightening␣hammer␣angle:", angle_at_measure_hammer)
print("Difference␣between␣hammer␣and␣camera:", angle_at_measure_hammer
    -(cam_tightening_angle))

plt.figure()
plt.plot(filtered_data['timeSeconds'], filtered_data['CamAngle'])
plt.title('Camera␣angle␣during␣a␣tightening␣full␣tightening␣cycle')

for t in filtered_data[filtered_data['state_change']]['timeSeconds']:
    state_int = filtered_data[filtered_data['timeSeconds'] == t]['
        OperationStateInt'].iloc[0]
    state_str = state_int_to_string(state_int, "operation")
    plt.axvline(x=t, color='grey', linestyle='--')
    plt.text(t, plt.ylim()[0], state_str, rotation=-45,
        verticalalignment='top', color='grey', fontsize=12)

plt.xlabel('Time␣(s)')
plt.ylabel('Cam␣angle␣(Degrees)')
plt.legend()
```

Appendix C Matlab Code

C.1 Stiffness Analysis

```
clear
%Biltema
T = [ 7.49 9.88 12.38]; %retightening torque to the same position
I_hammer = 2.8324*10^-3; %inertia of hammer
omega_hammer = [2.5 3.4 4] * 2 * pi; %angular velocity of the hammer before impact
K_socketanvil = T.^2./(I_hammer*omega_hammer.^2) %stiffness between hammer and screw

%%
clear
%TengTool
T = [7.09 10.16 12.74 ]; %retightening torque to the same position
I_hammer = 2.8324*10^-3; %inertia of hammer
omega_hammer = [2.45 3.2 4] * 2 * pi; %angular velocity of the hammer before impact
K_socketanvil = T.^2./(I_hammer*omega_hammer.^2) %stiffness between hammer and screw
%%
clear
%hex head
T = [11.57 21.72]; %retightening torque to the same position
I_hammer = 2.8324*10^-3; %inertia of hammer
omega_hammer = [2.5 4.06 ] * 2 * pi; %angular velocity of the hammer before impact
K_socketanvil = T.^2./(I_hammer*omega_hammer.^2) %stiffness between hammer and screw
```

C.2 Realisation 3

```
clear
close all
P = 0.00125;
D = 0.008;
d1 = 0.006647;
d2 = 0.007188;
N = 0.013;
dh = 0.009;
rm = (N+dh)/4;
alpha = pi*30/180;

my_g = 0.1;
my_u = my_g;

Cscrewansys = 1.1461*10^8; % N/m

Target_torque = [5 5 5 ...
                 5.5 5.5 5.5 ...
                 6 6 6 ...
                 6.5 6.5 6.5 ...
                 7 7 7 ...
                 7.5 7.5 7.5 ...
                 8 8 8 ...
                 8.5 8.5 8.5 ...
                 9 9 9 ...
                 9.5 9.5 9.5 ...
                 10 10 10]; %Nm
Target_torque_relubricated = [10 10 10];

Elongation = [50 48 45.5 ...
              58 59 59 ...
              65.5 61 63.5 ...
              67 73 69.5 ...
              64 66 66.5 ...
              73.5 71.5 72 ...
              72 73.5 71 ...
              78.5 77.5 79.5 ...
              84.5 78.5 86.5 ...
              86.5 79 85.5 ...
              91 90.5 92.5]*10^-6; %m

Elongation_relubricated = [107.5 94 99.5]*10^-6; %m

Tightening_torque = [5.2 5.1 5.3 ...
                     5.7 5.7 5.8 ...
                     6.4 6.1 6.4 ...
                     6.7 6.7 6.9 ...
                     7.3 7.3 7.3 ...
                     8 7.9 7.8 ...
                     8.1 8.3 8 ...
                     8.4 8.5 8.4 ...
                     9.1 8.9 9.3 ...
                     9.5 9.6 9.6 ...
                     10 9.8 9.9]; %Nm

Tightening_torque_relubricated =  [9.8 9.8 10.4]; %Nm

Loosening_torque = [3.3 3.3 3.2 ...
                    3.8 4.1 4 ...
```

```
                    4.5  4.1  4.4  ...
                    4.6  4.7  4.8  ...
                    4.9  5  4.8  ...
                    5.2  5.2  5.2  ...
                    5.8  5.9  5.8  ...
                    6.1  6.1  6.3  ...
                    6.8  6.4  6.7  ...
                    7.6  6.8  7  ...
                    7.1  7.3  7.2]; %Nm

Loosening_torque_relubricated = [6.6  7  7.4]; %Nm

Fp_torque = Target_torque/(P/(2*pi)+my_g*d2/(2*cos(alpha))+rm*my_u)*10^-3; %kN

Fp_tightuntight = (Tightening_torque-Loosening_torque)*pi/P*10^-3; %kN
Fp_Stretch = Cscrewansys*Elongation*10^-3; %kN

Fp_tightuntight_relubricated = (Tightening_torque_relubricated-
    Loosening_torque_relubricated)*pi/P*10^-3; %kN
Fp_Stretch_relubricated = Cscrewansys*Elongation_relubricated*10^-3; %kN
```

C.3 Matlab PLC code

```
clc,clear vars;
final_data = struct();

%PATH: Objects/PLC/Modules/<Default>/Global PV/visImpactVelocityMax
%opc.tcp://192.168.1.200:4840
serverList = opcuaserverinfo('opc.tcp://br-automation:4840');
uaClient = opcua(serverList);
connect(uaClient);

topNodes = uaClient.Namespace;
MaxV_T = findNodeByName(topNodes, 'visImpactVelocityMax', '-partial');
MaxHits_T = findNodeByName(topNodes, 'visImpactAmountLimit', '-partial');
MaxV_UNT = findNodeByName(topNodes, 'visUntightheningImpactVMax', '-partial');
MaxHits_UNT = findNodeByName(topNodes, 'visUntImpactAmountLimit', '-partial');
TagetAngle = findNodeByName(topNodes, 'visTighteningAngleTargetDeg', '-partial');
settings_nodes = [MaxV_T, MaxHits_T, TagetAngle, MaxV_UNT, MaxHits_UNT];

settings = readValue(uaClient,settings_nodes)
final_data.settings = settings

disconnect(uaClient);

% Define names for the new tables manually
new_table_names = {
    'CycleCounter', 'MainStateInt', 'OperationStateInt',...
    'HammerAngleDeg', 'MpAxisBasicPosition', 'MpAxisBasicVelocity',...
    'UntightheningAngleCurr', 'UntightheningImpactCounter',...
    'TighteningAngleCurrentDeg', 'TighteningImpactCounter','MotorTorque',...
    'CamAngle'
};

% Access the 'values' table inside 'signals' inside 'datalog'
values_table = datalog.signals.values;

[num_rows, num_columns] = size(values_table);

% Check if we have more data than variables
while num_columns > size(new_table_names)
    new_table_names = [new_table_names, 'UnknownVariable'];
end

% Create a struct to hold the columns
column_struct = struct();

% Loop through each column
for column = 1:num_columns
    % Extract data from the column
    column_data = values_table(:, column);

    % Assign data to the struct field with a manually set name
    column_table_name = new_table_names{column};
    column_struct.(column_table_name) = column_data;
end

final_data.impulsdata = column_struct

datetime = datetime()
% Save the struct to a MAT file in version 7.3 format
save('../../Testdata/24-04-26_NoScrewAnvil_1.mat', '-struct', 'final_data', '-v7.3');
```