Automatic testing of optical sight adjustment screws by a robotic arm

Jonas Gabrielsson

LUND
UNIVERSITY

Department of Automatic Control

MSc Thesis
TFRT-6190
ISSN 0280-5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Tryckeriet i E-huset
Lund 2023

Abstract

In this thesis the automation of an optical sight validation process was studied. The goal was to find a solution for controlling the position of a red dot in a red dot sight. A red dot sight is a non-magnifying reflector sight in which the user sees a red dot instead of a reticle. The movement of the dot was performed by fastening and unfastening adjustment screws on the sight. A side goal was to maintain the ability to find faulty sights, a task done by operators today. Two possible solutions were investigated, one using a controllable screwdriver with torque measurements, and one using a collaborative robotic arm. A tool-fitting strategy using force feedback was developed for the robotic arm. The torque reading and the red dot movement control was successful for both the screwdriver and the robot. The tool-fitting strategy performed very well for certain positions of the adjustment screw and about half of the times for the worst performing position.

Acknowledgements

I would like to thank Pontus Andersson and Lucas Rigestam at Aimpoint AB for the the support and the opportunity to write this thesis in collaboration with Aimpoint. A special thanks to Lucas for being positive throughout the project and for the motivation.

Björn Olofsson at LTH, I am grateful for valuable advice and feedback during the project.

Contents

1. Introduction **9**
- 1.1 Background . . . 9
- 1.2 Goals and questions . . . 11
- 1.3 Limitations . . . 12
- 1.4 Outline . . . 12
- 1.5 List of abbreviations . . . 13

2. Background **14**
- 2.1 Robot positions and frames . . . 14
- 2.2 Cyclic redundancy check . . . 16
- 2.3 Minutes of arc . . . 16

3. Adjustment by a screwdriver system **17**
- 3.1 Doga screwdriver system . . . 17
- 3.2 Testing beam . . . 19
- 3.3 Communication . . . 20
- 3.4 Setup . . . 21
- 3.5 Strategy . . . 23
- 3.6 Results . . . 25

4. Adjustment by a robotic arm **27**
- 4.1 Experiment setup . . . 27
- 4.2 Method . . . 29
- 4.3 Movement between the two adjustment screws . . . 30
- 4.4 The tool-fitting strategy . . . 30
- 4.5 Results . . . 38

5. Discussion **45**
- 5.1 The screwdriver . . . 45
- 5.2 The tool-fitting algorithm . . . 46
- 5.3 Tool-fitting strategy for the screwdriver . . . 48
- 5.4 Angle control . . . 49
- 5.5 Future work . . . 50

6. Conclusions **52**
Bibliography **53**

1

Introduction

1.1 Background

More and more small and medium sized companies decide to invest in automation. The benefits of automating tasks are many. Not only can it be profitable with the ability to scale up production and cut personnel costs, it can also free operators from monotonous and unergonomic tasks. Replacing previously manually performed tasks with fully or semi-automated solutions, can come with some challenges. As an example, it can be easy to miss defects and abnormalities previously caught by an experienced operator with a human eye and feel. A very common task that is present in both assembly and testing is the operation fastening and unfastening of screws. Locating screws and their orientation and fit tools to the them generates some challenges [Salem and Karayiannidis, 2020]. Even to get torque feedback to categorise if the screw is moving like planned can be an obstacle. Therefore much research is done in this field and some related articles are summarized below.

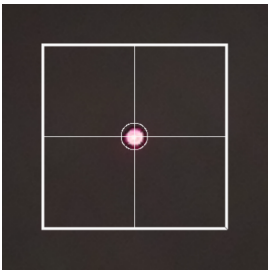
Apley et al. developed a method to categorise loosening of screws based on torque and angle measurements [Apley et al., 1998]. They developed an algorithm for detection of the four cases: The screw is loosening as planned, the tool is slipping against the head of the screw, the tool missed the screw head and the screw is stuck. The main focus of the article is to determine when the screw is slipping. This is done by comparing the periodic torque, coming from the tool trying to grab the screw head, with a periodic mathematical model.

Li et al. used a tool connected to a robotic arm to loosen bolts on a turbo charger [Li et al., 2020]. One problem that was studied was that of fitting the tool to the nut. The orientation of the nut is unknown so a pattern was developed for a rotating movement, and with the help of torque measurements they could then determine when the tool was fitted to the nut.

Salem and Karayiannidis developed a strategy to solve the hole-in-pin problem and extended it to the nut-in-screw [Salem and Karayiannidis, 2020]. They used a robotic arm with force and torque sensors to identify different stages during the application of the hole and the nut.

(a) The bit.

(b) A red dot sight.



(c) The adjustment area.

Figure 1.1 The bit used for testing (a), a compM5 sight(b) from Aimpoint AB and a screenshot(c) of the adjustment area.

Shauri et al. have with a two-armed robot screwed together a bolt and a nut [Shauri et al., 2012]. They used two cameras and force sensors. At a distance the position and orientation of the nut and bolt were decided by the cameras. When the two touch, their positions were estimated from the force sensors, and their positions were adjusted until the nut is fastened.

Aimpoint AB is a company whose goal is to automate their red dot sight validation process. A red dot sight is a non-magnifying reflector sight in which the user sees a red dot instead of a reticle. The validation is performed by moving the red dot along its specification range by turning the adjustment screws. In Figure 1.1a the bit used to turn the screws can be seen. The bit has two pins that go into two holes on the adjustment screws, which are the two metallic surfaces in Figure 1.1b. By turning the adjustment screws, the red dot inside the scope is moved up or down for the top screw as seen in Figure 1.2 or side to side for the screw on the side of the sight. The red dot should be able to move to the outer limits of its specifications without any abnormal pathing. The specification range is the square in Figure 1.1c. The process is today performed by an operator and is ergonomically heavy. With an automated process the storing of data and the traceability could be improved as well.

The process can be divided into four smaller parts, which are described in the following subsections.

Image processing

Tracking of the red dot in Figure 1.1c is performed by using a camera. The image must be processed in order to obtain signals for control and logging.

Loading

For the process to be fully automated the movement of sights to and from the testing rig must be done automatically. This is today performed by the operator.

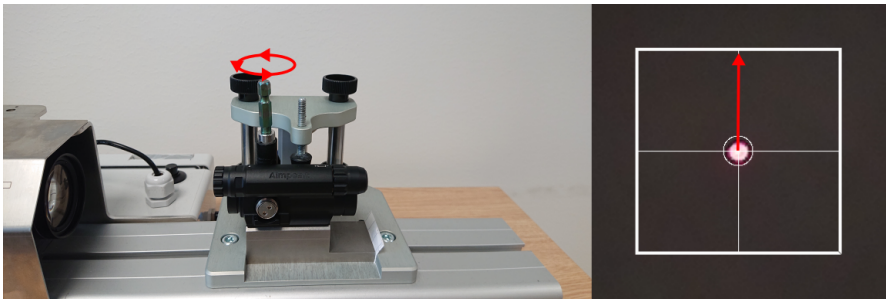


Figure 1.2 The red dot sight placed in the test rig with the bit placed on the top adjustment screw. Rotation of the top screw makes the red dot move along the y axis.

Testing

With the sight in the test rig the next step is to manipulate the adjustment screws. This will be done using a screwdriver and a robotic arm, respectively.

The goal of the test is to trace the outer edges of the adjustment range set by the sights specifications. This is done by fastening and loosening the adjustment screws. A strategy for fitting the tool to the screw is needed. While turning the adjustment screws the applied torque must also be monitored to make sure that the sight is not damaged but also to find any construction defects. The design of the adjustment screw gives a varying resistance and a "click"-sound, which must be accounted for in the control. The control of fastening and loosening the adjustment screws will first be implemented on a screwdriver with torque measurement and then a collaborative robotic arm.

Automatic validation

When the red dot has been moved according to the specifications, the sight has to be approved. This must be decided from the data obtained from the testing. Eventual torque or position anomalies must be taken into account. This can be done either automatically or by presenting the data to an operator, which then makes the decision.

1.2 Goals and questions

The goal of the thesis is to study how an automated screwing process can be developed and implemented. This is made by automating a validation process at Aimpoint. A tool-fitting strategy is developed inspired from previous work mentioned in Section 1.1, especially the strategies for finding the nut and the pin-in-hole. This thesis seeks to further investigate these strategies applied to small details and a screw head with a two hole layout. A consistent ambition is to work modular and having

in mind that the automated process can have subtasks performed by an operator in case of problems.

Questions to answer:

- How can a validation process for red dot sights be automated in a way that an operator can, if needed, take over some tasks?
- How is the head of the screw found and how can the tool be fitted to the screw using force feedback?
- How is a safe and robust control of the torque designed and implemented to ensure that the sight is not damaged?
- How can an automated solution be implemented, that still identifies defects previously found by an operator?

1.3 Limitations

This thesis focuses on developing solutions for moving the red dot using a screwdriver and a collaborative robotic arm. A strategy for fitting the bit to the adjustment screw with a robotic arm is to be developed. Automating the movement of the sight to and from the test rig and validation of the sights falls outside the scope of this thesis.

1.4 Outline

The main parts of the theory used in the thesis is summarized in Chapter 2. The first phase of the thesis is described in Chapter 3 and is performed with a controllable handheld screwdriver with torque measurements. The focus of this stage is to control the applied torque and move the red dot. Movement between the two adjustment screws is performed manually. In this chapter an evaluation of the torque data and the accuracy of the screwdriver is presented. The second phase of the thesis is focusing on fitting the tool to the two adjustment screws using a collaborative robotic arm with force feedback. This is described in Chapter 4. Discussion and conclusions are presented in chapter 5 and 6.

1.5 List of abbreviations

Table 1.1 A list of abbreviations

Abbreviation	Meaning
CRC	Cyclic redundancy check
MOA	Minute of Angle
PC	Personal computer
RPi	RaspberryPi
TCP	Tool center point

2

Background

This chapter presents some key parts of the theory used in the thesis. The first section explains how the position of the robot can be described and the two frames used in the thesis. The following section describes some key methods in the programming language used to control the collaborative robotic arm.

2.1 Robot positions and frames

The collaborative robot used in this thesis is a UR5e [Universal-Robots, 2021b] from Universal Robots. The robot is made out of rotating joints and tubes. The joints are numbered from 1 to 6 where the last joint closest to the tool is the sixth joint. The position of the robotic arm can be described in two ways. Either with a pose or with a joint configuration. The pose describes the position and orientation of the tool center point, TCP. This is done by three positions and three angles. The three positions are the coordinates in space expressed in the base frame. The base frame can be seen to the right in Figure 2.1. The three angles in the pose are how the TCP coordinate frame is rotated compared to the base coordinate frame. The robot's configuration can also be expressed in an array containing six angle values. This correspond to the angle values of each joint. Each of the joints is limited to one full revolution in each direction.

URscript

The robot is controlled over a TCP/IP communication with a PC and the robot accepts scripts written in its own programming language URscript [Universal-Robots, 2021a]. The sent scripts should be complete defined programs starting with the keyword "def" or "sec" and ended with "end". A few URscript commands used in the later implementation is described in the following subsections.

Threads. The URscript language have support for running several threads [Universal-Robots, 2021a]. A thread is defined in a program with the keyword "thread" followed by a name, the lines that should be executed in the thread and

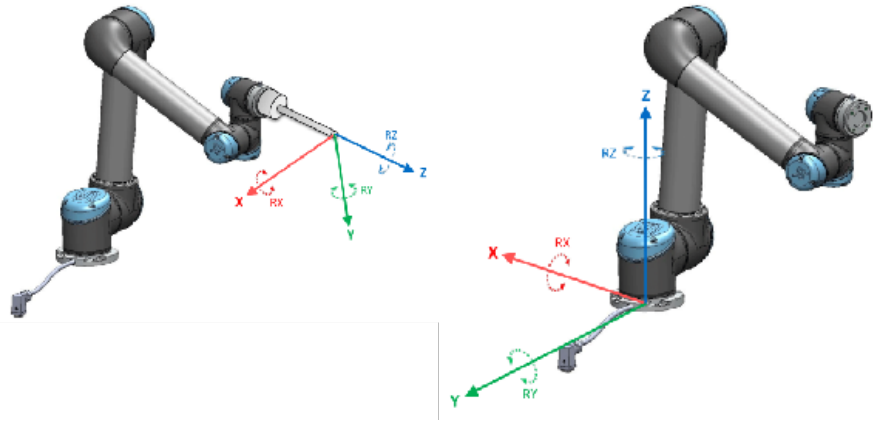


Figure 2.1 The TCP frame (left) and the base frame (right). The picture is taken from the UR5e user manual [Universal-Robots, 2021b].

then the keyword "end". With the thread defined it can be started and stopped with the commands "thrd = run threadname()" and "kill thrd". The controller is sending commands to the robotic arm with a frequency of 500 Hz. The time between these sends are given to the different threads for execution.

Interpreter mode. Instead of sending full programs, interpreter mode enables real-time control of the robotic arm [Universal-Robots, 2021a]. When interpreter mode is started a separate socket is opened on a new port (30020). This socket is listening for new commands sent as single lines of URscript. A line can contain several commands as long as the compiler can compile the entire string. The listening state is ended by sending the command "end_interpret_mode()" on the interpreter socket. In interpreter mode commands sent to the robot can be aborted by sending the command "abort" by itself on a single line. This is a way of stopping the robot while it is performing a task.

movej. The command movej has the syntax "movej(q,a=,v=,t=,r=)", where q is either a pose, describing the TCP pose, or an array of angles describing the joint configurations, a is the acceleration given in radians per second squared, v is a maximum velocity given in radians per second, t is the time the robot is given to perform the move, this overrides the acceleration and velocity values, and r is the radius of a blend given in meters [Universal-Robots, 2021a].

The blend gives the robot freedom to not exactly reach the target q, how far from q is given by the radius. This can be used in a series of move commands to give a smoother transitions between move commands. While using the movej command the robot control unit seeks to minimize the movement of the joints, this leads to non-linear movement of the tool [Universal-Robots, 2021b].

Force mode. In force mode the robot is no longer controlled by feeding it a desired configuration, it instead seeks to apply forces and torques to the TCP [Universal-Robots, 2021a]. The syntax for using force mode is force_mode(task_frame,selection_vector,wrench,type,limits). "task_frame" is provided by a pose that describes the target frame with respect to the base. The "selection_vector" is an array of ones and zeros that makes the robot able to move in the different axis of the frame. As an example, the array (0,0,1,0,1,0) makes the robot compliant in the z-direction and able to rotate around the y-axis in the provided frame. The "wrench" is a vector with the applied forces and torques in each direction. The forces are given in Newton and the torques in Newton meters. The "type" is a way of choosing modes, there are four modes available each transforming the provided task frame in different ways. The second is the one used in the thesis and it does not change the frame in any way. "Limits" is an array of six values with the allowed limits in each direction. If an axis is compliant the limit is for the allowed speed and if the axis is non-compliant the limit is the offset allowed before the force mode breaks for safety reasons. The force mode is ended by sending the command "end_force_mode()".

2.2 Cyclic redundancy check

Cyclic redundancy check, CRC, is used in different communication protocols, one of which is the Modbus protocol [Bies, 2021]. The CRC is a way of checking that the message has not been distorted during the transmission. The sent message gets a CRC value calculated and attached. When the message is received at the other end the same CRC calculation is repeated and the value is checked against the full message.

2.3 Minutes of arc

Minutes of arc, MOA is also known as minute of angle, especially in the marksman community, and is one sixtieth part of a degree. At 100 yards adjusting the sight one MOA means the point of impact is moved 1.0045 inches which is why it is suitable for American shooters. With American companies being a big part of the industry MOA is a commonly used unit.[*Minute and seconds of Arc* 2022]

3

Adjustment by a screwdriver system

The following chapter presents the screwdriver system and the testing beam and how it was used for red dot position control. It also describes the setup for the experiments and the strategy implemented for moving the red dot. The chapter ends with results from red dot position experiments.

3.1 Doga screwdriver system

Description

A screwdriver with torque measurement was provided by Aimpoint AB in the form of a system from DOGA. The system consists of one control unit and one screwdriver with torque measurement. The control unit was a DOGA MDTC-38 and the screwdriver a DOGA MDT2604-A [Doga, 2018b]. The control unit does not allow direct control of the screwdriver motor but instead works within a framework called presets. The screwdriver can be started and stopped by external communication. There is also the possibility to change values of parameters during run-time. There was room for 15 presets and every preset consisted of a series of parameters, e.g., desired torque, max torque, desired speed and more. When the screwdriver is used, an internal regulator tries to meet the parameters set by the used preset. The screwdriver and the control unit can be seen in Figure 3.1. In the picture the bit is mounted on the screwdriver. In this model the bit is floating, and can move slightly along the z-direction in a tool center point coordinate frame. The bit can also rotate when the motor is not engaged.

When the screwdriver is running the control unit has predetermined states, the states are "free speed", "free angle speed" and "fastening". What the controller seeks to achieve is dependent on this state and the parameters set. Depending on which state the control unit is in, the screwdriver can operate with different speeds and

Figure 3.1 The screwdriver and control unit used in the thesis.

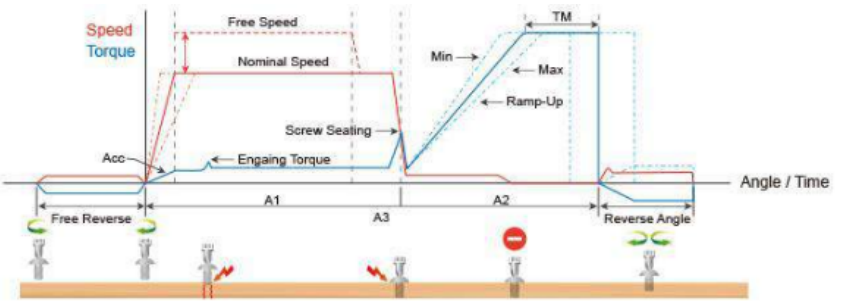


Figure 3.2 A graph showing the different states of screw tightening. The picture is taken from the Paramon manual [Doga, 2018c].

torques. The states are designed around the different stages of fastening a screw. A graph representation can be seen in Figure 3.2.

The first state is called free reverse and is an optional state. In this state the screwdriver can be set to drive in reverse a set amount of degrees. This is intended for fitting the tool to the screw head.

The second state is the free angle speed and is labeled A1 in Figure 3.2. In this state the speed parameter can be set in the range 0 to 1500 rpm. This state is ended by either too high torque or a that a set angle is achieved.

The third state is called fastening and is labeled A2 in Figure 3.2. This state can be triggered by torque rising, the rise in torque is meant to detect when the screw head is seated against the surface. In this state a new higher speed is set.

```
Run: b'\x01\x06\x0f\xa3\x00\x01\xbb<'
Change to preset 2: b'\x01\x06\x0f\xa4\x00\x02J\xfc'
Read values: b'\x01\x04\x0c\xe4\x00\x022\xac'
```

Figure 3.3 Three of the messages used in communication with the screwdriver control unit.

For instance, the regular speed parameter can be set in a range between 100 and 1500 rpm but the free range speed can be set in a range of 0 to 1500 rpm. So by operating in the free-range mode, the screwdriver can be operated at lower speed.

Communication with screwdriver control unit

Communication with the screwdriver control unit can be done in two ways, either via the serial port using the Modbus protocol or via the serial port using the free software ParaMON [Doga, 2018c]. The ParaMON software provides the possibility to set preset parameters and aquire and present real time data. This is not an option that can be used for real-time control.

The screwdriver is using the Modbus protocol and takes a message of a number of bytes and then responds according to the message [Doga, 2018a]. A baudrate of 115200 bits per second was used. An example of three messages that were used can be seen in Figure 3.3. The messages consists of eight bytes of hexadecimal data. The first byte is the device ID, followed by a byte choosing command; 04 means read and 06 means write. The following bytes state what memory address the command will affect and what should be written there. The last two bytes are for the cyclic redundancy check, CRC. The commands and their CRC:s were calculated beforehand and then sent when running the control script.

3.2 Testing beam

The testing beam consists of a pneumatic system for holding the sight in place. A camera is mounted at the beam, pointed through the camera. The camera is connected to a Raspberry Pi (RPi). The RPi has two major purposes. The original purpose is to provide visual feedback of the red dot to the operator adjusting the screws. The camera feed is shown on the screen and the adjustment range is placed on top of the feed as an overlay. The operator adjusts the screws and makes sure the red dot is following the desired route.

The second purpose of the RPi is to track the red dot and stream the coordinates to the PC. The streaming of the red dot coordinates is an extension of the original code provided by Aimpoint.

Figure 3.4 A red dot sight from Aimpoint, model compM5

Locating the dot

In the original RPi program, used by the company, a method for locating the red dot from the camera feed is implemented. The pixel with the highest intensity is found. Around this pixel a smaller image is produced, from the smaller image the center of gravity of the intensity is computed. This is then returned as the coordinates of the red dot.

The adjustment range

Every Aimpoint sight has a set specification of what adjustment range it handles. During this thesis all testing was made on a sight of the model Comp M5 as can be seen in Figure 3.4. The Comp M5 sight has an adjustment range of 2x2 m^2 or 69 MOA.

In the remainder of this thesis, the red dot positions will be expressed as pixels. The reference points, four corners and the center of the adjustment range seen in Figure 1.1c, were found manually by reading the pixel values when the red dot was placed at said positions. The Y axis is inverted, having zero at the top, with increasing pixels values moving downwards on the screen.

3.3 Communication

The RPi was connected to the PC via an UTP-cable. The red dot coordinates were streamed to the PC using the python class socket [Python Software Foundation,

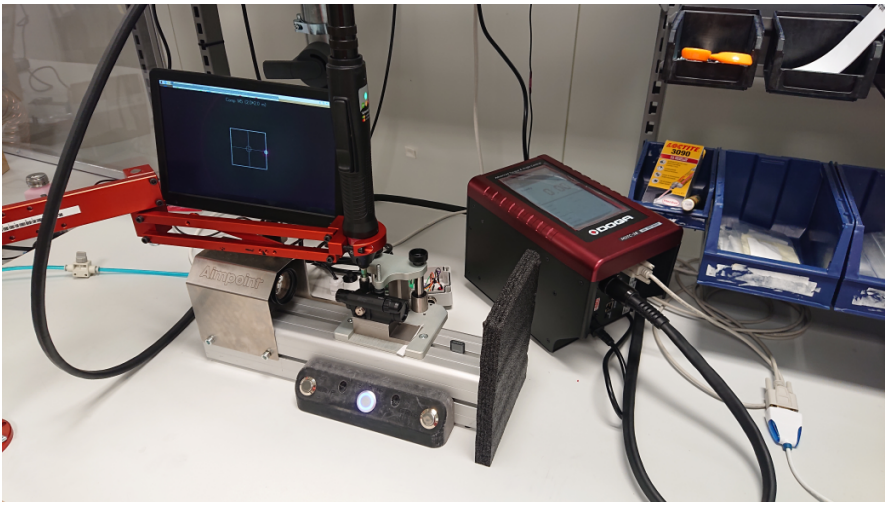


Figure 3.5 A picture of the physical setup in the red dot control testing.

2022b]. The RPi was sending coordinates non stop as a one way stream as fast as possible, to the PC which was reading the values.

3.4 Setup

Physical experiment setup

For the first stage of testing, the experiment setup can be seen in Figure 3.5 and a schematic figure can be seen in Figure 3.6 with the Part 1 configuration. A camera was connected to the RPi, the camera was capturing the movement of the red dot and the positions were computed and then sent to the PC. The RPi was also streaming to the screen mounted on the rig. The PC was both reading and writing values to the screwdriver control unit which was then controlling the screwdriver.

Red dot position regulator

The method of control was developed based on the options available from the DOGA system. Two presets were set up, one going in clockwise direction and one counter clockwise. Since the goal is to control the position of the red dot and not to tighten the screw, the presets were set up such that the entire operation of the controller was in the free range state. By operating in this state slow speeds was able as references for the internal controller. Since the screwdriver was only controlled by start, stop and parameter changes, the regulator implemented was simple. The screwdriver was sent a start command and ran until the position of the red dot was close to the desired position. Close to the reference the free angle speed was set to a

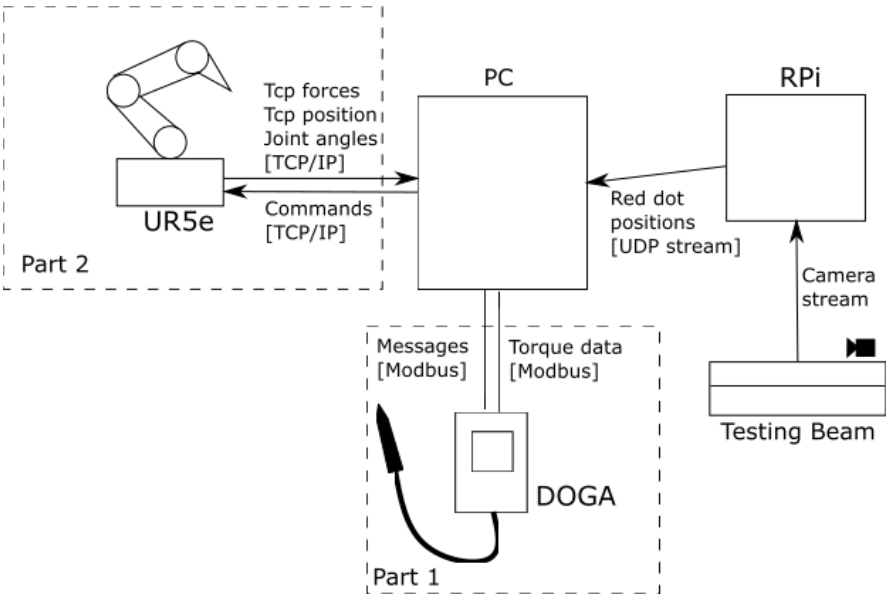


Figure 3.6 A schematic picture of the setup used in the red dot control testing. Part 1 shows the setup used for experiments with the screwdriver system and Part 2 the setup used for experiments with the UR5e collaborative robot.

low value. The screwdriver's speed was then reduced by the internal control. It kept running until one of three conditions were fulfilled:

- The measured torque was higher than a predetermined threshold.
- The position of the red dot was within a tolerance circle around the target position.
- The estimated next sampled position had overshot the target position.

The estimation of the next sample was as simple as taking another step with the same length as the last one:

$$\hat{x}[n+1] = k \cdot (x[n] - x[n-1]) + x[n] \tag{3.1}$$

where k is a regulator parameter for the step length, which was set to 1, and x is the position of the red dot sight in the present direction. The usage of the estimated next value was to prevent overshooting and enable the screwdriver to operate at higher speeds.

To minimize the issue of the low sample rate of the positions sampling of positions and torque was made concurrently using the python package asyncio [Python Software Foundation, 2022a]. This made the regulator loop time become similar to that of the position sample time.

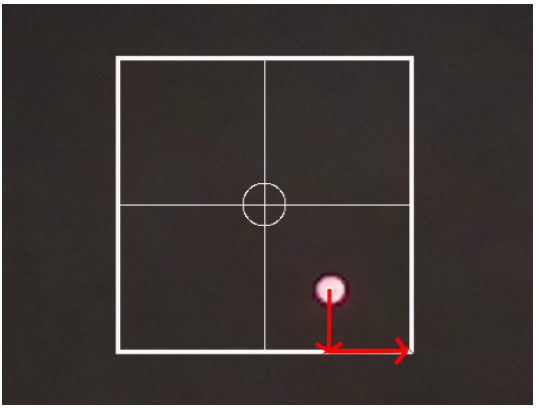


Figure 3.7 The path to the first corner for a starting point in the lower right quadrant.

3.5 Strategy

With the ability to control the position of the red dot, the next problem was to choose reference points so that the entire range was checked with as little movement as possible. The strategy for validating the adjustment range was to, from an unknown starting point, move the red dot to one corner. From there it should be checked that the red dot could trace the outer edges of the adjustment range and then the red dot should be placed in the center of the adjustment range. The main goal was to investigate if the square could be traced without fail. A secondary goal was to minimize the number of times the screwdriver had to move from one adjustment screw to the other, in order to prevent unnecessary movements. The strategy was divided into three stages, as described in the following subsections.

First corner

The red dot must be placed at the first corner from an unknown position. This was made by locating the red dot and identifying which corner it is closest to. The dot was moved out to the next side of the square in clockwise direction. To minimize the amount of movements between the screws, the first corner was always reached counter clockwise. This set up the next phase and the screwdriver does not have to change adjustment screw when the first corner was reached. An example can be seen in Figure 3.7. The bottom right corner is the closest, the dot is first brought down in y-direction and then moved to the corner.

Trace the square

The dot was now placed at a corner. At this stage the square was traced clockwise as can be seen in Figure 3.8. The screwdriver was always placed at the correct screw from the beginning. During the tracing, the movement of the "non moving" axis was

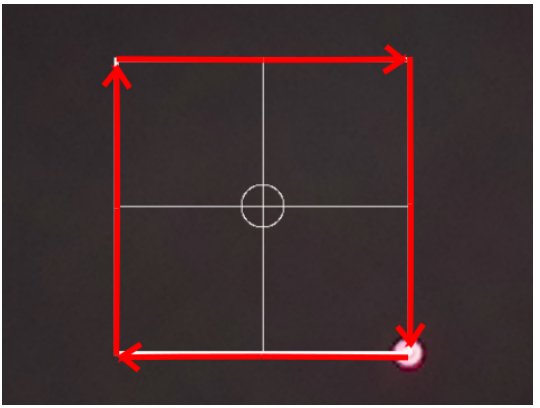


Figure 3.8 The adjustment range square is traced clockwise.

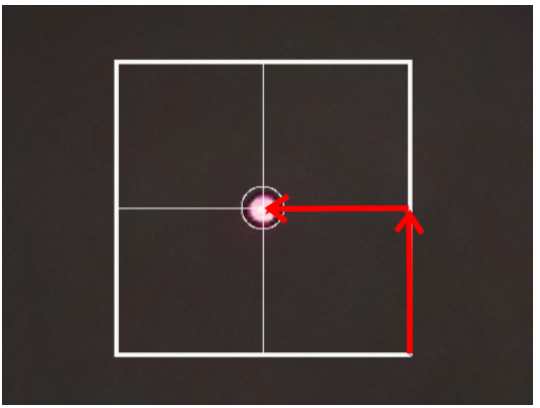


Figure 3.9 The path back to the center of the adjustment area.

checked for sway. Movement in the wrong direction would be a cause for the sight to fail the test.

Placing the dot at the center

When the red dot had reached the last corner the next step was to place the dot in the middle. Since the aim was to minimize the number of movements of the screwdriver, the first movement was always made along the same dimension as it just had moved in as seen in Figure 3.9. The dot was run to the center of a side of the square, then the screwdriver changed adjustment screw one last time and the red dot was moved to the center.

Under normal circumstances such strategy leads to six movements of the screwdriver during a test. This can be reduced by one movement if the starting position

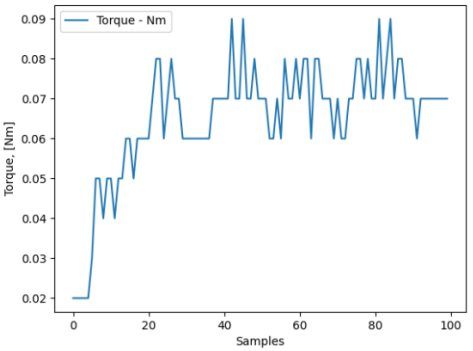


Figure 3.10 Torque data from a short test of fastening an adjustment screw. The data was sampled with a period of 0.015 s.

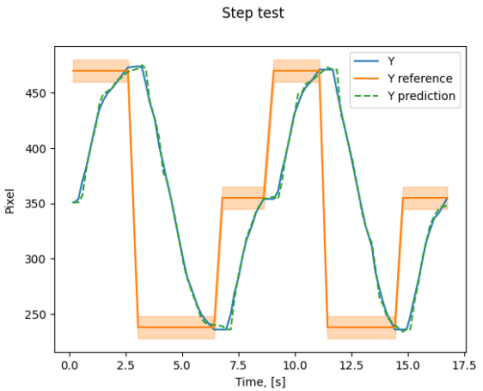


Figure 3.11 The result of a step response test in one direction.

happens to line up with a side, or two movements if the dot is located at a corner at the start.

3.6 Results

An example of the torque data for adjusting the red dot from the upper to the lower limit of the adjustment area can be seen in Figure 3.10.

The result of a step response test can be seen in Figure 3.11. There are two step lengths corresponding to the two step lengths that are necessary to fulfill the real test.

The plotted Y prediction was the next Y-step estimations but shifted one step

back. The shading around the reference is the tolerance that is used by the operators in the production today.

4

Adjustment by a robotic arm

The following chapter describes the steps taken to automate the validation process. It contains the movement between the adjustment screws using the strategy from the previous chapter, and an analysis of the measurements from the robot sensors. At the end of the chapter, a strategy for fitting the tool to the adjustment screw using only force feedback is presented and evaluated.

4.1 Experiment setup

The experimental setup can be seen in Figure 4.1, and a corresponding schematic in Figure 3.6 using the Part 2 configuration. The RPi unit and testing beam were the same as described in Section 3.4. The robot used was a UR5e collaborative robot [Universal-Robots, 2021b] equipped with a custom tool that can be seen in Figure 4.2. The tool holds the bit. The tool was designed with two goals in mind. Firstly the tool needed to be stiff to obtain a reliable force feedback from the bit touching the environment, and secondly the tool needed to have some length to put some distance between the robot and the testing beam. This was to increase the movement possibilities close to the testing beam where the access is limited and the pointed design increased this further. During the testing of the movement strategy, a modified sight was used for the adjustment screw. The sight was modified to have a 4 mm hexagonal head for easier tool-fitting. The modified sights can be seen in Figure 4.3. During the evaluation of the fitting algorithm an original compM5 sight was used.

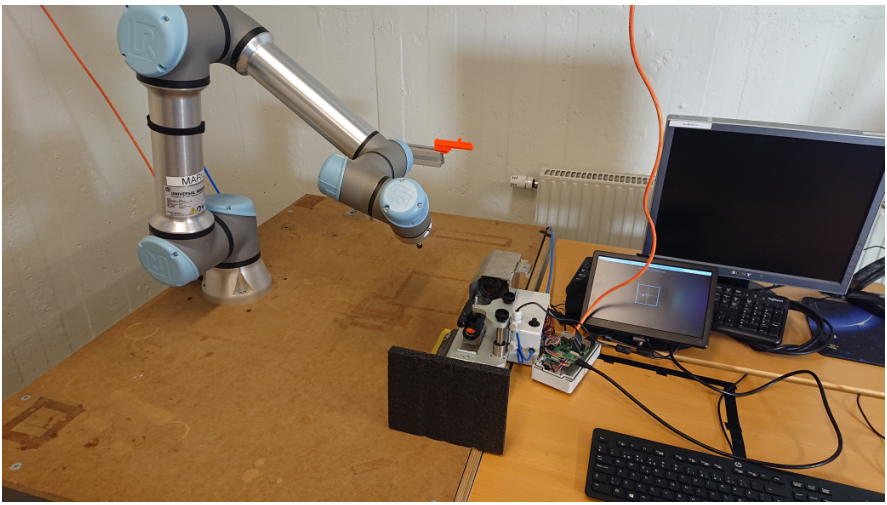


Figure 4.1 The experiment setup showing the robotic arm, the testing beam and a compM5 sight.

Figure 4.2 The UR5 robot with the custom tool holding the bit.

(a) The modified sight **(b)** The original sight

Figure 4.3 The two sights used in the experiments.

4.2 Method

Controller architecture

The system consists of three major parts; the UR5e robotic arm, the PC running the controller and testing beam with the integrated camera and RPi. Communication between the PC and the robot was made via TCP/IP [Johansson, 2022]. The robot was running in interpreter mode with a thread running in parallel that sends the data to the PC. The data were sent as strings containing three arrays written sequentially. The first six values corresponded to the measured TCP force along x, y and z and the torques around these vectors. The following six values are the position of the TCP and the last six values are the robot's configuration described by its joint angles.

The program running on the PC consisted of three threads running in parallel. One main thread contained the strategy and sent commands to the robot. One thread read the stream of positions from the RPi and one thread received and unpacked the strings from the robotic arm. The communication threads are updating state variables and when the main thread needs data it reads from these variables. To avoid that the variables are read and written at the same time, the main thread locked the variables while reading from them. The control loop in the main thread has a sample rate of 50 Hz.

Preprocessing of measurements

The torque and force data extracted from the robotic arm were often offset in some way. To remove the offset from the data a method was implemented that sampled 20 values, then calculated and returned the mean value. The data showed signs of some drift. To minimize the impact of the drift the offset was removed between different tasks performed by the robot.

During movement of the red dot the torque data for joint six were used to avoid too high torque on the adjustment screw. The torque data showed some specific behaviour when the robot applied torque to the joint. Each time the motor engaged, a large spike was seen. It then settled at one specific offset for each direction. To remove the effect of the spike a few samples were not acted upon directly after the motor engaged. Two separate offset values were produced experimentally and removed based on direction.

4.3 Movement between the two adjustment screws

The movement between the two adjustment screws was designed with the strategy from Section 3.5 in mind. The robot needs to have a home configuration to start from and depending on the red dot position move to the appropriate adjustment screw for the next red dot movement. Five angle configurations were and found recorded using the tablet connected to the robot. Two configurations were above the top adjustment screw, one was tilted for the original sight and one for the modified sight. Two similar configurations were found for the side screw. Finally, a home configuration was placed away from the testing beam. Movement between the configuration was obtained with the movej command. The home configuration was used as a transition configuration for movements between the top and side screw by using a blend with $r = 0.05$. To know when the robotic arm had completed its movement a routine was implemented to check if the current configuration matches. To test the strategy without having to take the tool-fitting into regard the modified sight was used and the bit was exchanged for a 4 mm hexagonal bit. A simple tool-fitting strategy was implemented. The starting position was straight above or besides the adjustment screw. The tool was then moved against the screw until contact was measured in the z-direction by using force control. When the tool was in contact with the screw, a torque was applied while the tool was pushed against the screw. The tool was guided by the chamfered hole on the screw down into the hole. When red dot movement was received the robot was stopped and the tool was assumed to be seated in the hole.

4.4 The tool-fitting strategy

In the following section the algorithm for the tool-fitting is described as executed on the top position. The pins on the bit will throughout the chapter be named as lower pin, being the one closest to the surface of the adjustment screw in the starting position and upper pin, the one furthest from the surface in the starting position. The starting position can be seen in Figure 4.4. The left of the pins is the lower one. The algorithm is executed in the same way on both the top and side adjustment screw.

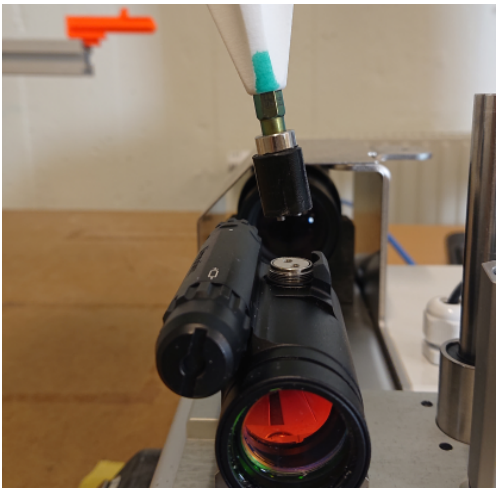


Figure 4.4 The tool in the starting position of the fitting strategy.

During the execution of the algorithm, the actions taken were decided by the state of which the robot was in. The algorithm consists of the following states: starting position, push, sweep, seat the pin, confirm hole, swirl and tilt up, a diagram of a state graph can be seen in Figure 4.8. The different hole configurations will be referred to by their capital letter and can be seen in Figure 4.10. The hole on the left half circle will throughout the section be mentioned as the first hole. When movement is described in a coordinate frame the coordinates are in the tool center point coordinates, unless anything else is specified. The algorithm seeks to fit the tool to the adjustment screw. For a perfect fit the two pins are seated in the two holes and the surface of the tool planted against the surface of the adjustment screw.

Starting position

There were several reasons for the robotic arm to be placed in the starting position state. It might be that the robot was moved there from either the home or side position. It can also be because the algorithm has failed in one of its states and is moved here as a reset. To leave the state, the joint angles of the robots are checked. When the angles correspond to the angles of the desired position it transitions to the next state. The starting position of the tool can be seen in Figure 4.4. The tool is tilted about 20 degrees from the center.

Push

The goal of the initial stage of the algorithm was to ensure that the lower pin was being in contact with the surface of the adjustment screw. From the starting position the tool was pushed forward using force mode with a force in the z-direction. The state can be exited in one of two ways. The robot either connects with something

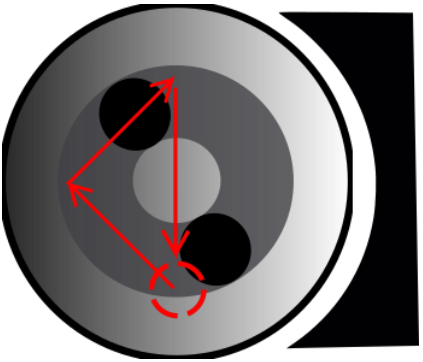


Figure 4.5 The desired sweeping pattern in search of the first hole.

and measures a force in z-direction or it timeouts without receiving contact, the former leading to a transition to the next state and the latter moving the tool back to the starting position. The tilt of the tool ensures that when contact was made with the adjustment screw it was always with the lower pin. The desired point of impact can be seen in Figure 4.5, marked as a dashed ring. The force threshold used to detect contact is 1 N.

Sweep

With the first pin in contact with the adjustment screw, the next stage was to sweep the surface for the first hole. The desired search trajectory can be seen in Figure 4.5 showed with red arrows. The darker shading indicates the possible hole locations. The initial hole configuration was unknown but thanks to symmetry sweeping one side of the adjustment screw was enough to make sure that the opportunity was given to detect one hole. The trajectory was divided into three parts and to follow each part forces were applied to achieve motion along each direction. The distance travelled was calculated from measured positions of the tool center point, TCP. The distance travelled was used as breaking points for when the algorithm sends a new force command to the robot.

The detection of the hole was made by force and torque readings. A constant force was pressuring the pin against the surface and as the pin moved, the forces and torques in x- and y-directions were monitored. A spike in the measured data indicated that the pin had made contact with either a hole or something else. Each recognized contact was treated as a potential hole and the robot was transitioning to the next state. The calculated travelled distance for the hole detection was used as an estimate of the hole location and was passed along to later states. The thresholds used for hole detection were 1.8 N for the forces or 0.5 Nm for the torque.

Force mode, described in Section 2.1, is based on the coordinate frame for the tool center point. To transform these forces into the coordinate frame of the adjustment screw the 2D rotational matrix was used. As seen in Figure 4.6 the coordinate

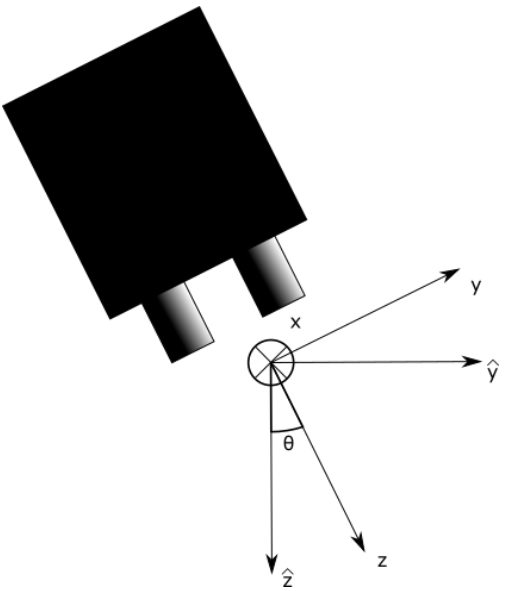


Figure 4.6 The tool's coordinate frame x, y, z and the adjustment screw coordinate frame $\hat{x}$, $\hat{y}$, $\hat{z}$.

frames are separated by a rotation θ.

To achieve the desired movement in Figure 4.5 with some force still pushing down in $\hat{z}$-direction, the following forces were applied, given as (x,y,z) in N: For the first leg (1.3, -2.5, 2), the second leg (1.3, 2, 2), the third leg (-1.3,$-2\tan\theta$,2). The applied force in the y-direction in the last leg was to compensate for the tilted angle of the tool. The force applied in z-direction while the pin was in contact with the surface resulted in a sliding motion in the $\hat{y}$-direction. This was the reason for the higher applied y-force in the first leg compared to the second leg.

Seat the pin

When the robot had reached the "seat the pin" state the pin was either in a hole, on the edge of a hole or had received a false hole detection. The aim of this state was to, if possible, seat the pin better in the hole. This was made by pushing it downwards towards the surface of the adjustment screw. The tool was allowed to move in x- and y-directions and the applied force together with the tapering of the hole guided the pin into a more secure fit. The force was applied in a short pulse. The robot could leave the state in two ways depending on the difference in position of the pin. If the pin moved too much when the force was applied, the pin did not seat but slide on the surface. If the pin movement was small, the robot transitioned to the next state under the assumption that the pin was seated. The applied force was $(0,-2\tan\theta,2)$ N in a 0.5 s pulse.

Confirm hole

The confirm hole state was used as a confirmation that the pin was actually placed in a hole and not stuck on some other place at the adjustment screw. In this state the movement along x- and y- direction is constrained. The robot tried to move forward and backward in both x- and y-direction. The forces in these directions were monitored and to pass the check, enough reactive force must be measured. If all four force direction checks were passed the conclusion was that the pin was in a hole and the robot transitioned to the next state. As soon as one check was failed the robot transitioned back to the starting position.

Swirl

Upon reaching the swirl state the lower pin was assumed to be seated in a hole. The exact location of the hole was unknown but an estimate was made from the travelled distance until the initial hole detection. Depending on the estimated hole location two different strategies were applied. The goal of both strategies was to locate the other hole with the upper pin. This was made by reducing the tilt (the angle θ) of the tool, while rotating it. Pivoting around the seated lower pin to ensure that the upper pin traversed the second hole. This state was left either with two connected pins resulting in red dot movement or with no detected movement of the red dot which lead to the conclusion that the two pins were not connected.

If the hole was located close to the "middle", positions B, C and D in Figure 4.10, the slim version of sweep was applied. The tilt of the tool was reduced by applying a force in z-direction while allowing the tool to rotate around its x-axis. With the first pin seated this created a resulting torque around the x-axis. When the second pin came in contact with the surface the tool was no longer able to rotate around the x-axis. The applied force was then delivered to the screw and the measured reactive force in z-direction was increased. When the second pin was in contact with the surface the second phase of this state began. The tool was rotated in clockwise or counter-clockwise direction for the upper pin to seat in the second hole. The direction of the rotation was decided from the estimated distance travelled in the sweep for the first hole. A travelled distance between 2.5 and 5.5 mm triggered the slim swirl. Detection of both pins being connected to the holes was done by observing the movement of the red dot. With bots pins connected the applied torque rotated the screw and then moved the red dot. In the rare case that second hole lined up perfectly with the upper pin the second pin was seated immediately when the tilt angle was reduced. When the tool was rotated the red dot moved and the algorithm stopped.

If the first hole was located away from the middle, position A or E in Figure 4.10, the original sweep algorithm was applied. The approach was altered slightly from the slimmer behaviour. To reduce the tilt of the tool there was a rotating torque applied. The tool was allowed to move in x- and y-direction and rotate around the x-axis. With the first pin seated in the hole the applied torque pivoted the tool and

reduced the tilt angle. When an increasing force in z-direction was detected the second phase was initialized and the tool was rotated. By initiating the rotation from the start, the angle that the upper pin needed to rotate while the second pin was being in contact with the surface was reduced.

In the original swirl the robot was allowed to move in x- and y-direction and rotate around the x- and z-axis. The applied torque was 1 Nm. For the slimmer swirl the robot was allowed to move in z-direction and around the x-axis. The applied force was 0.5 N and there was a torque applied around the x-axis of 0.2 Nm. This was not enough to rotate the tool but it prevented it from falling. Both the original and the slim enters the second phase, which was rotating while pushing the tool against the surface. This was done by allowing movement in all directions and rotation around the z-axis. A force of 0.5 N was applied in z-direction and 1 Nm of torque was applied around the z-axis. For both the original and slim version, the direction of rotation was decided by the travelled distance to the first hole, a travelled distance of less than 3 mm resulted in counter clockwise and more than that clockwise. The red dot had to be moved 3 pixels before the two pins were seen as seated.

Seat both pins

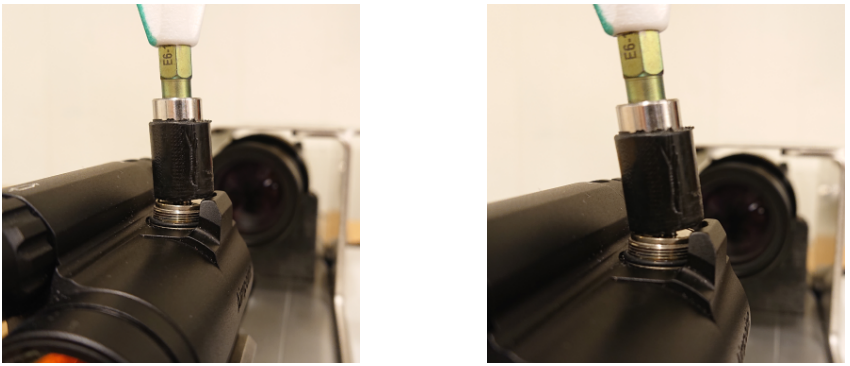
In the final state both pins were connected but there was no guarantee for the tool being connected properly. There were mainly one or two of the following issues: The pins were not seated all the way down in the holes leaving a gap between the surface of the tool and the surface of the adjustment screw. Otherwise the tool was tilted and therefore prevented a perfect fit. Examples of these cases can be seen in Figure 4.7.

To tilt up the tool and seat both pins the robot was tasked to move to a known configuration by using the movej command. The robot was sent a joint configuration of the robot being connected with both pins in the holes but with the tool placed slightly above the surface of the adjustment screw. The joint 6 value was updated with the current joint angle of joint 6 to avoid the last joint from moving and the pins to twist out of the holes. When the robot had moved to the new position the two pins were still connected and the tool was no longer tilted. A small force was applied in z-direction using force mode to seat the pins and connect the surface of the tool to the surface of the screw.

Angle control

Movement of the red dot was made by rotating the adjustment screws. With the tool fitted to the screw the robotic arm could be tasked to move joint 6 to a specific angle. A factor between rotated angle of the adjustment screw and red dot movement was derived by comparing known points and the measured angle between them:

$$\alpha_1 - \alpha_2 = y_1 - y_2 \implies \frac{\alpha_1 - \alpha_2}{y_1 - y_2} = 1 \tag{4.1}$$



(a) Floating **(b)** Tilted

Figure 4.7 Two common positions for the tool when both pins are connected to the two holes.

where α_1 and α_2 are angles in degrees and y_1, y_2 are pixel values of the red dot as seen in Figure 4.9. Converted to radians the following was obtained:

$$e \cdot \frac{y_1 - y_2}{\alpha_1 - \alpha_2} \cdot \frac{2\pi}{360} = \Delta\beta \tag{4.2}$$

where e is the error in position and $\Delta\beta$ is the calculated angle in radians. The red dot positional error was calculated and the current joint angles were read. The robot was then tasked to move to its current joint configuration but with the desired step, $\Delta\beta$, added to the joint 6 angle. During the movement the torque of the last joint was observed and if the torque was too high the task was aborted to prevent damage to the sight. The torque threshold used during the experiments was set to 0.3 Nm. This threshold was set based on the discarded sights used during the development process and will need to be updated after testing on functional sights. The movement was executed using the command movej with the following parameters, $a = 1$, $v = 2$, $t = 0$ and $r = 0$ as described in Section 2.1.

Evaluating the tool-fitting algorithm

The performance of the algorithm was evaluated for five different hole configurations. The configurations were chosen to cover the five most interesting points in the left half plane of the adjustment screw. The positions can be seen in Figure 4.10 and are labeled from A to E. The hole configurations were chosen as the most challenging configurations for the sweep and the swirl actions. Positions A and E were chosen because they are the endpoints and the swirl would have to rotate the furthest in each direction to find the second hole. The B and D position were chosen because they were placed at the border for when the swirl transitioned from the original to its slimmer behaviour. Close to the D position was also where the swirl needed to change its rotation direction.

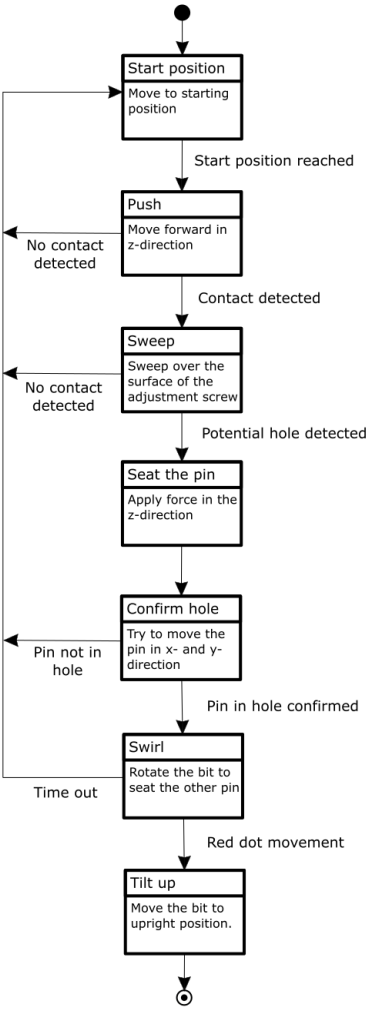


Figure 4.8 A state graph for the tool-fitting algorithm.

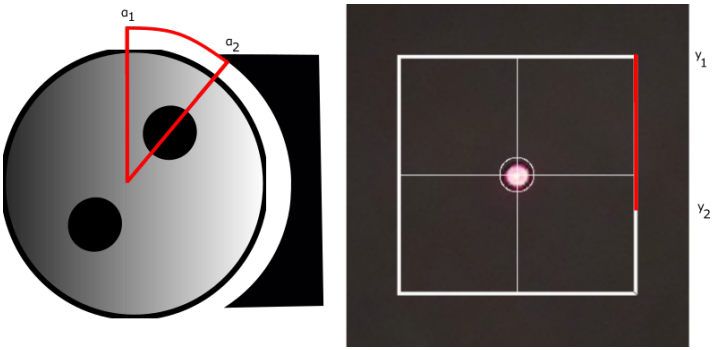


Figure 4.9 α and y in calculations of the factor between rotated angle and red dot movement.

Figure 4.10 The five hole configurations used in evaluating the algorithm.

4.5 Results

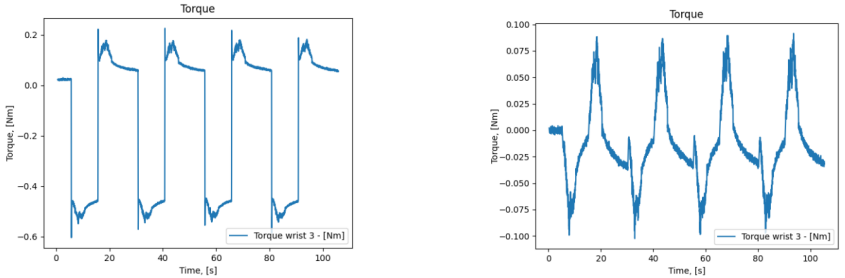
Preprocessing of measurements

The robot was tasked to move joint 6 in total 60 degs clockwise, then back to the starting position. The torque data from joint 6 can be seen in Figure 4.11a. The experiment was performed without any external load and repeated four times during the run. In Figure 4.11b the result can be seen from the removal of spikes and offset. Note the y-axis compared to the unprocessed data in Figure 4.11a.

The tool-fitting algorithm

The tool-fitting algorithm was evaluated in two different ways. One experiment was performed to emulate the algorithm used in a production setting and the other experiment was designed to evaluate the different parts of the strategy separately.

During the experiment designed to replicate a production environment the robot was moved from the home configuration to the starting position. The robotic arm was then tasked to perform the tool-fitting strategy. If the outcome was a success, the adjustment screw was turned between 30 and 180 degs at random using the angle control method. If the two pins were seated in the holes the outcome was marked as a success. The robot was then returned to the home position and the experiment was



(a) Raw measured data. **(b)** Offset and spikes removed.

Figure 4.11 Measured torque for joint 6 during a repeated 60 degree step test.

Table 4.1 Results from the separate evaluation.

	A	B	C	D	E
Sweep	20/20	20/20	20/20	20/20	20/20
Seat first pin	5/5	5/5	5/5	10/10	5/5
Swirl	20/20	16/20	12/20	15/20	20/20

repeated. The results were 14 succeeded attempts of 30 possible. This experiment will be referred to as the production experiment.

The results of the separate evaluation of the main parts of the tool-fitting algorithm can be seen in Table 4.1. The table shows the number of successful tests for each hole configuration for the three main parts of the tool-fitting algorithm. What constitutes a successful experiment and further details about the test conditions are explained in each section.

Sweep. To test the sweep part of the algorithm, the robot initialized in the home configuration. It was moved to the start position above the top adjustment screw and the "push" was initialized followed by the "sweep". What constituted a successful run for the sweep was to first make contact with the adjustment screw surface, then sweep the screw and stop when a hole was detected. The pin did not need to be seated in the hole, just remain in contact with it. After a hole was detected the robot went back to the home configuration and the experiment was repeated. The experiment was performed 20 times for each hole configuration. During the experiments with hole configuration A, four of the twenty hole detections were made of the hole close to the point of impact. The other detections were made at the end of the search pattern.

The measured forces and torques during a successful run can be seen in Figures 4.12 and 4.13. The figure is from an experiment with hole configuration B.

Seat first pin. The testing of the first pin seating was performed by extending the previous test with the seat first pin method. A successful seating of the pin results

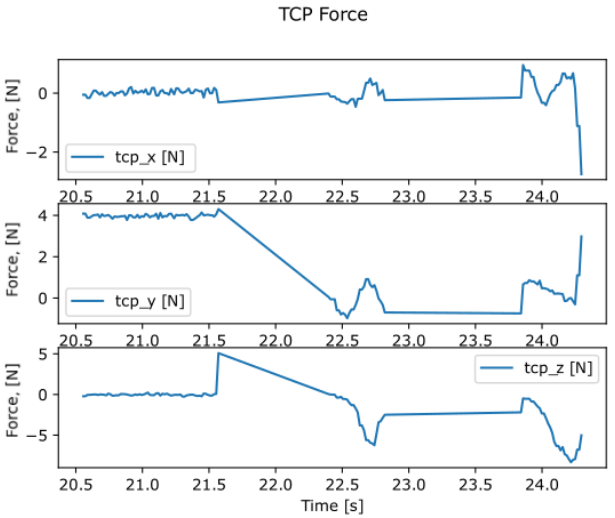


Figure 4.12 Forces for a typical run during the push and sweep for the first hole.

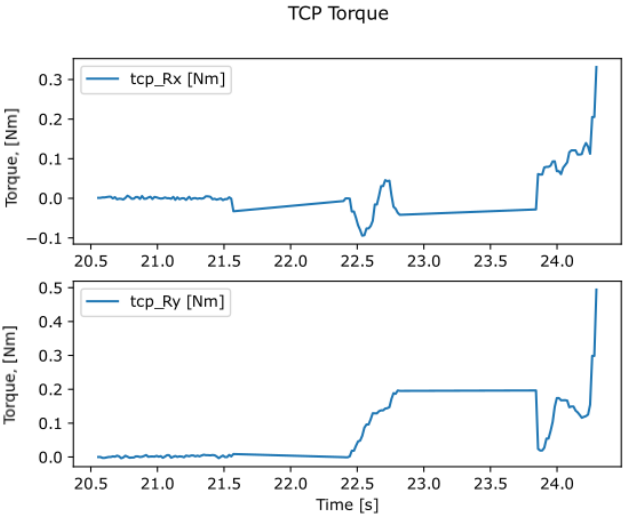


Figure 4.13 Torques for a typical run during the push and sweep for the first hole.

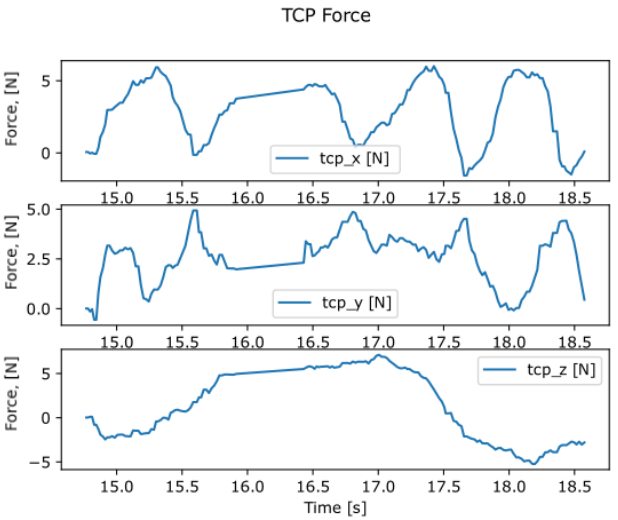


Figure 4.14 The measured forces during the seating of the second pin using the original swirl.

in the pin being moved from around or in the hole to a secure seated position down in the hole. Since the seating does not change depending on the hole configuration the test was repeated five times for each configuration. An exception was made for the D configuration as seen in Table 4.1. The reason for this was that the pin here often was located at the edge of the hole and not seated in the hole after the sweep. This made the seating of the first pin extra challenging.

Swirl. The evaluation of the swirl method was made by letting the robot start in the home position and then apply the tool-fitting strategy. The experiment was repeated 20 times for each hole configuration. The method assumes the first pin to be seated properly and all attempts presented in Table 4.1 had a proper seated first pin. The hole configuration of the adjustment screw was reset between each attempt.

The force and torque data of a typical seating of the second pin using the original swirl can be seen in Figures 4.14 and 4.15. The red dot position during the fitting of the second pin can be seen in Figure 4.16.

Angle control

The results of a step experiment can be seen in Figure 4.17. The x and y coordinates can be seen in the upper and the lower figure. The angle control was tasked to take a 60 pixel step from 310 to 370 in the y-direction.

Torque data from a 60 pixel step experiment can be seen in Figure 4.18. There was a 60 pixel step starting at 5 s, and then a step in the other direction at 15 s. Another experiment was performed to test if the method aborts the movement if the

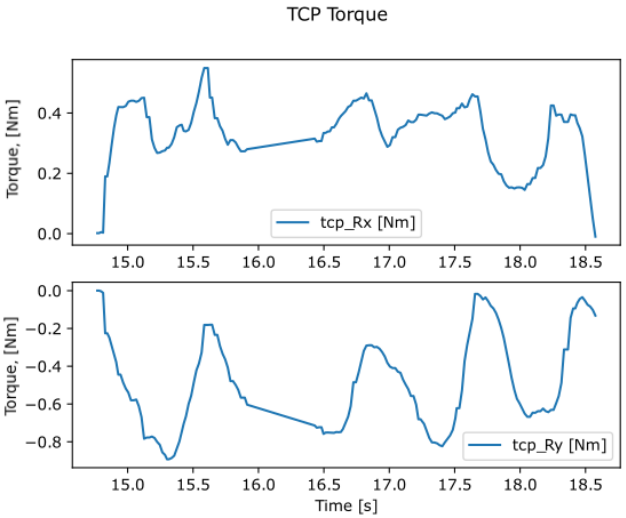


Figure 4.15 The measured torques during the seating of the second pin using the original swirl.

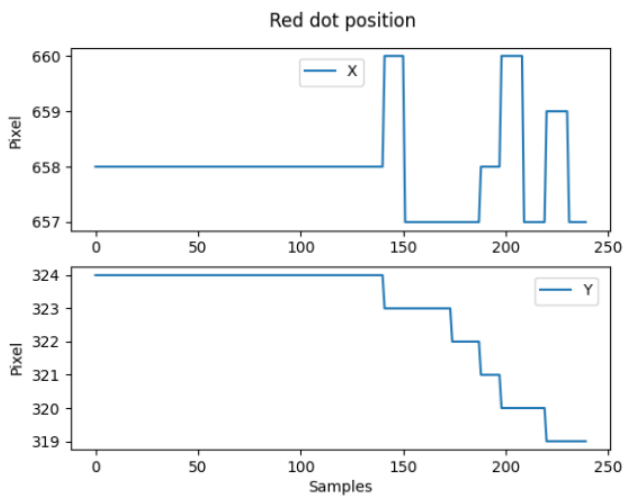


Figure 4.16 The red dot movement during the swirl. The sample period was 0.02 s

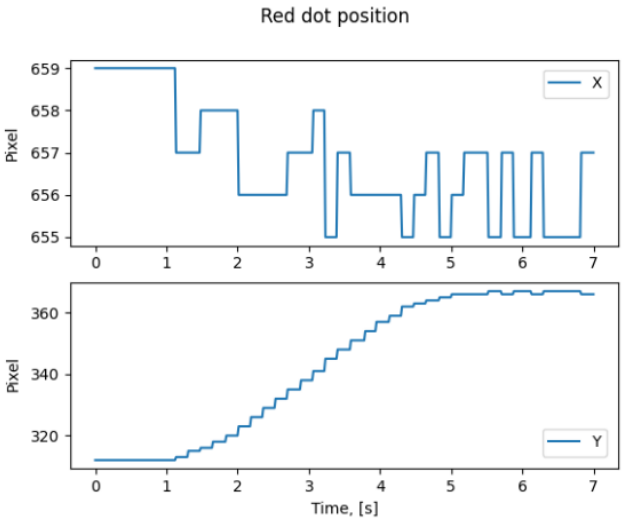


Figure 4.17 A 60 pixel step test in the y-direction.

torque was too high. The time response of torque can be seen in Figure 4.19. The task was to, after 3 s, turn the adjustment screw one revolution. The sight used had worn out adjustment screws and a too high torque was detected and the program aborted.

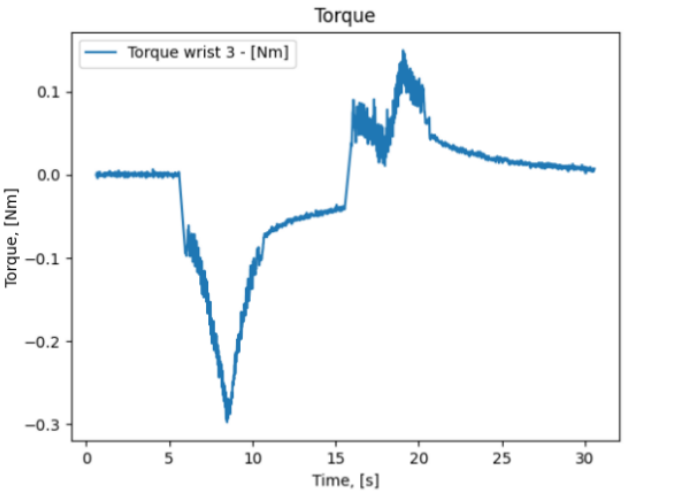


Figure 4.18 Torque data from a double step test during angle control.

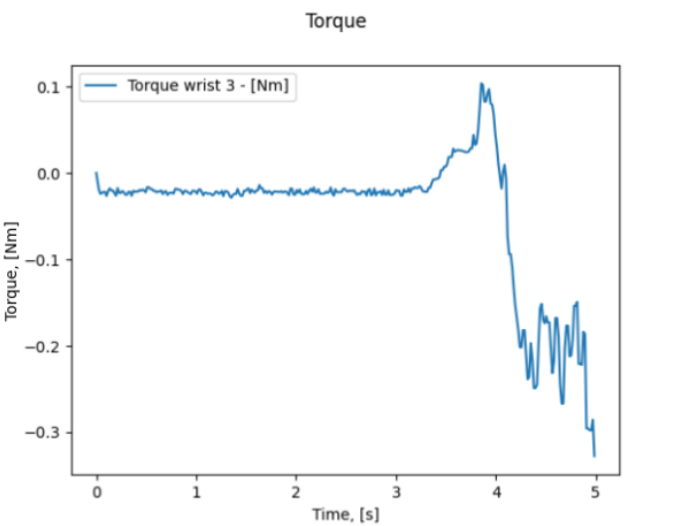


Figure 4.19 Torque data from a step test on a faulty sight, the algorithm aborted due to significant high torque detection.

5

Discussion

In this thesis one possible way of automating the screwing process was studied and a solution was developed and implemented. Due to limitations in the accuracy of the torque data from the screwdriver the method of categorizing the state of the screw of Apley et al. could not be implemented and tested [Apley et al., 1998]. A strategy for fitting the bit to the adjustment screws was developed inspired by the work of Li et al., Salem and Karayiannidis and Shauri et al. [Li et al., 2020] [Salem and Karayiannidis, 2020] [Shauri et al., 2012]. The solution was developed in a modular fashion and the different parts of the algorithm can be further developed separately. The tool-fitting strategy can for instance be replaced by another one if the model of the sight is changed. The developed solution was implemented on a collaborative robot and the testing beam is unaltered so it is possible for an operator to interfere and perform subtasks.

In the following sections, the performance of the screwdriver, the tool-fitting strategy and the red dot movement control are discussed.

5.1 The screwdriver

The angle control regulator performs satisfactory. There is a slight overshoot sometimes as can be seen in Figure 3.11, but it is well within the margins and to the human eye the red dot is on the line.

Adjusting the screws of the sights requires about 0.1 Nm. The resolution of the torque measurements is 0.01 Nm. This leads to quantization of the torque data as seen in Figure 3.10. The low resolution of the torque data eliminates the possibilities of using the torque data for anything more advanced than checking a high limit. Aimpoint is trying to detect sights with too high torque and this can be done using this screwdriver setup.

The DOGA system is not built for real-time control of the screwdriver motor. The motor is controlled by the internal regulator of the control unit. A user can only command the unit to start and stop, this limits the possible control strategies. This device is probably more geared towards assembly with an operator and predefined

tasks. The presets can then be programmed to work well in specific tasks. The goal here is to control a process in real-time and for this it would be easier to bypass the internal regulator and control the screwdriver motor directly.

A solution for automating the process by using the screwdriver without a robotic arm could be to by using two screwdrivers build a bigger rig with the two screwdrivers placed one on each adjustment screw. The fitting of the screwdrivers could be made by a operator that then starts a script that controls the screwdrivers. The next step is to make the screwdrivers fit the tools by giving them 1 degree of freedom so they can engage and disengage. This solution would be very dependent on the screwdrivers being placed exactly correct. The rig would also be very hard to use for sights of different sizes.

5.2 The tool-fitting algorithm

In the experiment where the algorithm was used from start to finish the score was 14 succeeded attempts out of 30. This seems low compared to the scores of the different states evaluated individually. This is definitely a case of where the chain is as strong as its weakest link. The most challenging part of the strategy is the swirl state where depending on hole configuration the success rate was between 50 and 100%. With the algorithm used only for the worst performing hole configuration, C, the expected success rate should be about 50%. The 30 attempts performed were made on different hole configurations. The design of the experiment leads to more attempts being taken on the harder hole configuration because of the fact that the screw is only rotated if the strategy was successful. This was seen in the data as many failed attempts in a row before a successful attempt and rotation of the hole configuration. This is one explanation for the score being lower than expected from data in Table 4.1. This is, however, not a fault in the experiment method since during real application the robot would have to try several attempts on the hole configuration before giving up or requesting help.

Another difference between the production test and the separate testing of components is that the swirl method always started with a good fit of the lower pin in the first hole during the separate testing. In the production test this was not always the case. This is another possible reason for the strategy underperforming in the production experiment.

An issue with the setup that was detected during the production test was the movement of the test beam. This comes from the tilt up action being too aggressive in combination with the testing beam not being properly secured to the table. This induced movement of the testing beam between attempts. There is a risk that this affected the result.

Sweep

As seen in Table 4.1, the sweeping part of the algorithm performs with high accuracy. The performance comes from the data and how much a hole detection separates itself from the pin just sweeping along the surface. In Figure 4.12 the force data from the push and sweep can be seen. The contact is clearly seen in the lowest figure around 21.5 s. After the contact, the program sleeps for a second before starting the first leg of the sweep. The first leg is completed at around 22.7 s. One second later the second leg of the sweep is started. This is ended with a hole detection as can be seen clearly with spikes in both the force data in x- and y-direction as well as in the torque data in Figure 4.13.

Seat first pin

The seating of the first pin works as intended, if the pin is in the hole the method simply seats it further and if the pin is on the edge the pin is moved into the hole with the help of the chamfered holes. During the testing of the D configuration, the sweep left the pin on the edge of the hole. The testing was therefore extended to further evaluate the performance of the seating in the edge case. A case that did not show up during the testing was when the sweep gives a faulty hole detection or the pin stops at the very edge of the chamfer. The seating then has a risk of pushing the pin along the surface of the adjustment screw and not into the hole. This was tested separately by moving the testing beam to make the hole detections worse. During these tests the seating succeeded 3 out of 6 times, the failed attempts was successfully detected by the measured distance.

Swirl

The performance of the method for seating the second pin is very dependent on the hole configuration as seen in Table 4.1. For the outer configurations the method is reliable but for the inner there is a drop in performance. At configuration C about 50 percent of the attempts were successful. A probable reason for this comes from the position of the tool compared to the holes and the way pressure against the surface is applied. In the slimmer variant the tool is in a more upright position so the applied torque around the z-axis gives more rotation of the tool instead of pressure against the surface. To compensate for this, tries were made with higher force applied in z-direction. This instead tended to make the tool balance on the second pin against the surface and rotate the first pin out of the hole. As the first hole moves away from configuration C the performance increases, the angle of the tool becomes greater and the second pin is more likely to seat in the hole. There is a special case between hole configuration C and D which is where the tool needs to change rotation direction. The outcome in this area is very dependent on the measured distance from the sweep. A rotation in the wrong direction gives the robot zero chance to fit the second pin. Hole configurations B and D show comparably good performance, the issue here is that both the slim and the original swirls are at

the location where they perform the worst. Which one that is chosen is depending on the measured distance from sweep.

The test result from the swirl test was checked against the programs' log files to find cases of false positives. During the experiment one false positive was found from the 17 failed attempts. In this particular attempt the second pin jumped out of the hole at the same time as the robot stopped. A false positive could lead to the angle control starting and the surface of the adjustment screw being scratched.

The seating of the second pin is the most challenging task in fitting the tool to the adjustment screw. This can also be seen in the results in Table 4.1. The swirl method works very well for the outer hole configurations but struggles a little with hole configurations B to D. The main issue with the fitting of the second pin is to identify when the pin is in the hole. There is no clear way to identify from Figures 4.14 and 4.15 that the pin is seated. It would be natural to find the hole detection in the torque data from joint 6. Unfortunately, when applying torque the measured data look like in Figure 4.11b and the small differences from a hole detection is undetectable.

The movement of the red dot is used as a stopping condition as seen in Figure 4.16. As both pins are connected the applied torque rotates the screw and moves the red dot. As seen earlier the measured position in x-direction moves a few pixels up and down when the red dot is moving. To prevent false positives the threshold for pixel movement must allow some wiggle room. The sacrifice for preventing false positives is that the sensitivity is lowered. This leads to some missed attempts for the slimmer swirl, especially where the two pins were in the holes and rotated the screw but then jumped up before the red dot moved enough pixels to break.

5.3 Tool-fitting strategy for the screwdriver

To integrate the screwdriver into the sweep and swirl strategy some alterations must be made. With the strategy so dependent on force feedback the measured forces must be evaluated. The screwdriver used in the thesis had the bit floating and it was able to rotate while the screwdriver motor was not engaged. This will probably lead to some issues. The floating bit might make the seeking for the first hole harder since the bit might be in contact with the surface before any forces are measured. The bit's ability to rotate freely might make the swirl fail. The swirl part is dependent on the pivoting around the seated first pin. To test the strategy for the screwdriver it needs to be mounted and the robot needs to be calibrated for the new weight and offset. With a proper calibration and a detailed new TCP, the code should function with the screwdriver attached except for the swirl method. Where the robot for the custom tool rotates the bit applying torque to joint 6, the rotation of the bit must instead come from starting the screwdriver. Developing an alternative to the swirl step might be necessary. The configuration for starting points and home will have to be mapped out again.

The simplified fitting algorithm used in the evaluation of the movement between

screws can easily be implemented for the screwdriver. A preset can be set up for the screwdriver in which it rotates slowly. When the screwdriver is in contact with the adjustments screw, the screwdriver starts to rotate while the robot is pushing gently against the surface. When red dot movement is received a stop command is sent to both the screwdriver and the robotic arm. This fitting strategy can not work for the original compM5 sight but for other models that have other heads with chamfered holes on the adjustment screws it might be worth implementing and evaluating. Another way of implementing this is to design adapters and in that way alter the head from the two pins to an easier shape, much like how the sight was modified in the testing. The automation problem is then made simpler at the expense of another part that needs to be designed and tested. This falls outside of the scope of this thesis. Testing with the screwdriver on the modified sight was cut because of time constraints.

5.4 Angle control

As seen in the lower plot in Figure 4.17 the angle control performed well. The angle control method uses the command movej and the internal angle control of the robot is sufficiently accurate to perform this task. The data displayed in the figure is just one of many tests and the result was often in the range of 3 pixels from the said reference. This is a more accurate result than the implemented angle control for the screwdriver system. The screwdriver used 5 pixel tolerance for a run to be evaluated as successful. Both the angle control of the UR robot and the screwdriver are performing to a standard that satisfies the goals set for this thesis. The demands from the company is that the red dot can come in contact with the line. This translates to a pixel threshold of about 10 pixels. The step length during the evaluation was set to 60 pixels. This results in slightly less than 360 degrees rotation of joint 6. This is because of the UR5e joint 6 angle limitations. To move the entire adjustment area a minimum of 235 pixels are required. To apply this method of angle control the robotic arm needs to have unlimited rotation of the last joint.

In the upper figure in Figure 4.17 the red dot position can be seen. Since the top screw was the one being used the red dot should ideally not move at all. The movement seen for the x-axis can be a result of a few things. It is not rare that the position data move between two values. This happens when the intensity center of the red dot is located between two pixels. Noise in the camera stream can then make the intensity lean towards one pixel in one sample and towards the other in the next. If there is a more then one pixel movement along the wrong axis there is probably either movement of the sight in the rig or the sight is not working properly. During the experiments, the test beam was flooded in strong light because of other experiments in the room. This light pollution can also be a possible reason for the moving x-position data. To prevent the light from affecting the camera the testing beam was shaded (to the best of author's abilities).

The measured torque during adjustment of the screws is best shown in Figure 4.18. In this figure there is also a relic from the data processing appearing. At time 15 s when the angle control takes a step back to the original position, two spikes can be seen in the torque. This looks out of place compared to the behaviour at second 5 or compared to data from the screwdriver in Figure 3.10. This is the remnants of the great spike that shows up in the data at the start when torque is applied at joint 6.

As seen in Figure 4.19 the method can be aborted if the measured torque is too high. This is important to prevent damage to the sights during testing. The measured torque from the TCP-sensor is sufficiently accurate for aborting the function if needed. After the varying offsets and spikes were removed, seen in Figure 4.11b, the data can be checked for high torque values. The angle control method was tested and developed on discarded sights and the adjustment screws tend to become harder and harder to turn with more experiments. The torque threshold of 0.3 Nm that is used throughout the thesis is probably set too high for functional sights fresh of the production line. The threshold will have to be adjusted after testing on actual sights.

A potential issue with the torque data is the removal of the offset. In Figure 4.11b after the step, the value seems to stabilize around -0.05 Nm. This indicates that the offset for that direction is not exact enough. The offsets were derived from test results and can vary from day to day. If a new sight takes torques around 0.1 Nm (as seen in Figure 3.10) to turn the adjustment screw an error of 0.05 Nm is a major issue. This is of course also dependent on what torque threshold is set, since the main function of the torque reading is to avoid too high torque. If this was to be implemented in a factory setting, the removal of the offset needs to be automated and evaluated further.

5.5 Future work

To improve the performance of the tool-fitting strategy certain aspects can be investigated. Since the tool-fitting strategy is meant to be used for moving the tool to both adjustment screws in an alternating fashion the position of the holes for both screws can be remembered. After the first tool fit on the screw, the hole configuration is known and will not change unless the robot is there. This can be used as a complement to the existing strategy or used to make a new more efficient one. The robot can also be fitted with additional sensors, e.g., a vision system to find the positions of the holes. With the hole configuration known the strategy can be simplified or a brand new strategy can be developed.

The main part to improve is the method for fitting the second pin. The swirl is performing the worst for hole configurations B to D. Here, there is probably an alternative method that can be more successful. Another way of solving the issue with the slim swirl is to rotate the tool and the robot around the hole to make the conditions similar to the case where the original swirl performs well and use the

original swirl. This would demand more space around the sight and might not work for the current setup.

Before the sweep and swirl method can be implemented using the screwdriver the limitations of the force feedback must be evaluated. Further development and modifications are probably needed to make it work.

6

Conclusions

The thesis considered mainly three ways of automating the process of validating red dot sights. The first alternative uses two screwdrivers, one for each adjustment screw, and will require a custom-built rig for each sight. This solution can be fully automated or used together with an operator fitting the tools and starting the testing. The second way is to employ a robot for performing the screwing. Advantages with this solution are that advanced strategies can be implemented and the force feedback is reliable. Disadvantages are the cost and complexity of a robotic arm with potentially additional sensors. The third way is to mount the screwdriver on a robotic arm. An advantage with this solution is that there are no demands on the angle limit of the last joint. The robotic arm just needs to be able to carry the screwdriver while still having accuracy. A disadvantage with mounting the screwdriver is that complex fitting strategies become harder to perform.

Fitting the tool to the adjustment screw can be performed using only force feedback as the results show. The performance of the strategy must be improved before implemented into any kind of production. The strategy is modular and potential further research can be conducted for just the swirl which is the worst performing step.

The screwdriver and the robotic arm had both the sensors and control necessary to move the red dot according to the desired specifications. The data received are of high enough standard to ensure that the robot or screwdriver is not damaging the sights.

The way of finding anomalies and defects comes from the different measurements used in the process. By logging the red dot movement and torque data faulty sights can be identified. By logging the red dot movement and the time, potential slips in the screw can be detected. With enough data samples of working and faulty sights, research on a classifier for automatic validation might be possible.

Bibliography

Apley, D. W., G. Seliger, L. Voit, and J. Shi (1998). "Diagnostics in disassembly unscrewing operations." *International Journal of Flexible Manufacturing Systems: Design, Analysis, and Operation of Manufacturing and Assembly Systems* **10**:2, pp. 111–128. ISSN: 0920-6299.

Bies, L. (2021). *CRC calculations*. URL: https://www.lammertbies.nl/comm/info/crc-calculation (visited on 2022-07-11).

Doga (2018a). *COM protocol MODBUS*. URL: https://www.doga.fr/sites/doga/files/uploads/documents/60307%5C_1.pdf (visited on 2022-07-11).

Doga (2018b). *Instructions manual*. URL: https://www.doga.fr/sites/doga/files/uploads/documents/60429.pdf (visited on 2022-07-11).

Doga (2018c). *Paramon manual*. URL: https://www.doga.fr/sites/doga/files/uploads/documents/60304.pdf (visited on 2022-07-11).

Johansson, A. T. (2022). *Ur_py_ctl*. URL: https://ur-py-ctl.readthedocs.io/en/stable/reference/ur_py_ctl.html (visited on 2022-10-03).

Li, R., D. Pham, J. Huang, Y. Tan, M. Qu, Y. Wang, M. Kerin, K. Jiang, S. Su, C. Ji, Q. Liu, and Z. Zhou (2020). "Unfastening of hexagonal headed screws by a collaborative robot." *IEEE Transactions on Automation Science and Engineering* **17**:3, pp. 1455–1468. ISSN: 1545-5955.

Minute and seconds of Arc (2022). Accessed: 2022-12-09. URL: https://en.wikipedia.org/wiki/Minute%5C_and%5C_second%5C_of%5C_arc%5C#Firearms.

Python Software Foundation (2022a). *Python asyncio documentation*. URL: https://docs.python.org/3/library/asyncio.html (visited on 2022-08-17).

Python Software Foundation (2022b). *Python socket documentation*. URL: https://docs.python.org/3/library/socket.html (visited on 2022-10-03).

Salem, A. and Y. Karayiannidis (2020). "Robotic assembly of rounded parts with and without threads." *IEEE Robotics and Automation Letters* **5**:2, pp. 2467–2474. ISSN: 2377-3766.

Shauri, R., K. Saiki, S. Toritani, and K. Nonami (2012). "Sensor integration and fusion for autonomous screwing task by dual-manipulator hand robot." *Procedia Engineering* **41**, pp. 1412–1420. ISSN: 1877-7058.

Universal-Robots (2021a). *The URScript Programming Language for e-Series*. Document version: 9.3.84.

Universal-Robots (2021b). *Universal Robots user manual*. Document version: 9.3.116.

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER'S THESIS

Date of issue
February 2023

Document Number
TFRT-6190

Author(s)
Jonas Gabrielsson

Supervisor
Pontus Andersson, Aimpoint Aktiebolag, Sweden
Lucas Rigestam, Aimpoint Aktiebolag, Sweden
Björn Olofsson, Dept. of Automatic Control, Lund University, Sweden
Yiannis Karayiannidis, Dept. of Automatic Control, Lund University, Sweden (examiner)

Title and subtitle
Automatic testing of optical sight adjustment screws by a robotic arm

Abstract

In this thesis the automation of an optical sight validation process was studied. The goal was to find a solution for controlling the position of a red dot in a red dot sight. A red dot sight is a non-magnifying reflector sight in which the user sees a red dot instead of a reticle. The movement of the dot was performed by fastening and unfastening adjustment screws on the sight. A side goal was to maintain the ability to find faulty sights, a task done by operators today. Two possible solutions were investigated, one using a controllable screwdriver with torque measurements, and one using a collaborative robotic arm. A tool-fitting strategy using force feedback was developed for the robotic arm. The torque reading and the red dot movement control was successful for both the screwdriver and the robot. The tool-fitting strategy performed very well for certain positions of the adjustment screw and about half of the times for the worst performing position.

Keywords

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
1-54

Recipient's notes

Security classification

http://www.control.lth.se/publications/