CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Cutting Tool Wear: Data Acquisition and Automation of Wear Measurements on End Mills

Master's thesis in Computer science and engineering

Pontus Andersson

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2022

Master's thesis 2022

Cutting Tool Wear:
Data Acquisition and Automation of Wear Measurements on End Mills

Pontus Andersson

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2022

Cutting Tool Wear: Data Acquisition and Automation of Offline Wear Measurements on End Mills
Pontus Andersson

Supervisor: Yinan Yu, Computer Science and Engineering
Advisor: Ana Bonilla and Edvard Svenman, GKN Aerospace
Examiner: Aarne Ranta, Computer Science and Engineering

Master's Thesis 2022
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Gothenburg, Sweden 2022

Cutting Tool Wear: Data Acquisition and Automation of Offline Wear Measurements on End Mills
Pontus Andersson
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Tool management has an important impact on machining operations. As cutting tools are used, they are worn out and thus they need to be changed. The cost of cutting tools is a significant cost of a machining operation. Likewise using a tool too long can damage the produced part. To balance these two factors the wear on cutting tools must be analyzed and measured. Manual measurements take a long time and cost a lot of money, thus an automatic process for measuring cutting tool wear would be beneficial. Many different approaches for automatic tool wear measurement have been researched. This work proposes a method for automatic wear measurement of end mills using computer vision. The method uses a line scan camera and rotates the end mill to collect an image of the entire tool circumference. A two-step approach was then used to analyze the images, the first stage used a Mask R-CNN model to identify the worn areas. After that, the worn area was quantified by measuring the maximum and average depth of flank wear and the maximum fracture dept. Four Mask R-CNN models with different hyperparameters were trained. The methods were tested on three types of end mills all used for machining super-alloys. The best model resulted in a mean absolute measurement error of 13 % for average flank wear depth, 16 % for maximum flank wear depth, and 24 % for maximum fracture depth. To investigate the inclusion of new tools, experiments were also done on one end mill type that was not used when training the Mask R-CNN model. This resulted in an average error of 48 % for average flank wear depth, 40 % for maximum flank wear depth, and 48 % for maximum fracture depth. The results are promising, however, more research is needed to evaluate how this could improve tool utilization and machining processes in an industrial environment.

Keywords: computer science, cutting tool wear, end mill wear, deep learning, neural network, image analysis

Acknowledgements

During my thesis, I received great support. I want to give a big thanks to my supervisors Ana Bonilla, Edvard Svenman, and Yinan Yu for their continued support throughout the project. Without your time and expertise, this work would never have been possible.

I would also like to thank all the colleagues at GKN Aerospace who have helped me along the way and made the time at GKN thoroughly enjoyable. It has been a pleasure working with all of you.

Finally, I would like to thank GKN Aerospace for giving me the opportunity to write a thesis that combines machine learning and machining. Both of which are areas that I am interested in.

Pontus Andersson, Vänersborg, June 2022

Contents

List of Figures

List of Tables

Glossary

2D Two dimensional. 3
3D Three Dimensional. 3

ADC Analog to Digital Converter. 30, 34

bbox Bounding Box. 44

CNN Convolutional Neural Network. 20, 21, 26
CoC Circle of Confusion. 14
COCO Common Objects in Context. 40, 44

FCN Fully Convolutional Network. 20, 21, 37
FCNN Fully Connected Neural Network. 19, 20
FN False Negative. 18, 44
FP False Positive. 18, 44
FPN Feature Pyramid Network. 37

GPU Graphics Processing Unit. 26

IoP Intersection over Prediction. 44
IoU Intersection over Union. 18, 44, 47–49, 51, 53, 59
ISO International Organization for Standardization. 2, 24

RGB Three Red, Green and Blue. 39
ROI Region Of Interest. 21, 37, 38, 47
RPN Region Proposal Network. 21

T10 The 10 mm end mill in Fig. 3.2b. 23, 34–36, 43, 46, 47, 50, 53–55
T12 The 12 mm end mill in Fig. 3.2c. 23, 34–36, 43, 46, 47, 50, 53–55
T8 The 8 mm end mill in Fig. 3.2a. 23, 34–36, 43, 46, 53–55
TN True Negative. 18
TP True Positive. 18, 44

WSN Wear Segmentation Network. 43, 44, 47–56

1
Introduction

1.1 Background

High quality and fine tolerances are vital for the function, safety, and efficient operation of many components, an example is critical jet engine parts. One method for producing parts with fine tolerances is machining, where cutting tools are used to remove material to shape the final part. The wear of the tools has an impact on the quality of the part that is produced [1]. Thus, tools need to be changed regularly to ensure a good quality of the final part.

According to [2] cutting tool costs can be as high as 8 % of the total manufacturing cost and up to 34 % if machine downtime and poor product quality are taken into consideration. A typical cost of cutting tools for machined components is 3 % of the total component cost [1]. Optimizing the utilization of tools is therefore important to minimize the cost of machined components. Furthermore, [3] showed that increased tool life can allow for a reduction in energy consumption of up to 12 % and a decrease in the production cycle time of up to 15 %. Hence, increasing tool life is an important factor to reduce both the cost and waste of machining processes.

To maximize tool utilization without damaging the workpiece, it is important to know how long a specific tool can be used. There are different methods of determining tool life and they can be split into two main groups: model-based and sensor-based. The model-based methods use known parameters of the machining process to predict the tool life. Examples of models are Taylor's model which uses the cutting speed, chip thickness, and depth of cut to determine tool life, [4] and Usui's model which has normal pressure on the tool face and tool face temperature as input parameters [5]. One thing the models have in common is that they depend on some unknown constants. These constants are specific to each machining setup, thus the models require manual calibration before they can be used. The calibration is done by milling the test parts while continuously measuring the tool wear.

For the sensor-based methods, tool wear is instead measured. This can be done both outside the machine as well as in the machine during the machining process. When it comes to measuring tool wear during the machining process, there are two different approaches: direct measurement and indirect measurement [6]. Direct measurement is based on directly measuring wear, thus having the advantage of capturing tool geometry and how it changes as the tool wears off. Methods used for

direct measurements can be based on optical sensors, vision systems, radiation, and more. However, direct measurements can be hard to carry out during the machining process because the harsh environment in the machine makes it hard to capture the wear. For instance, both the workpiece and coolant can obscure the view of the tool. This is why indirect measurements are important, the idea of indirect measurement is to predict the tool wear based on other measured parameters. Examples of parameters used for indirect measurement are cutting force, temperature, vibrations, and acoustic emissions. Correlating the indirect measurements with tool wear requires manually measuring the tool wear and calibrating the models.

All methods used to estimate tool life require a method of directly measuring tool wear. This is clear in the case of direct measurement techniques. In the case of indirect and model-based approaches, direct measurement is required for calibration. Currently, tool wear is most commonly measured by examining the wear manually using a microscope [1]. This is time-consuming and severely limits the amount of data that can be collected. According to [1] it requires approximately 40 hours of machining to create tool life curves for two cutting speeds using the ISO turning test [7, 8]. Thus, a way of automatically measuring tool wear could greatly help the process of collecting data on tool wear. Automatic wear measurement could also help with quality control. This since part quality is correlated with cutting tool quality, thus if the tool wear is known, it is possible to estimate imperfections in a part. Again this is time-consuming since every tool used for producing a part will need to be analyzed for wear.

1.2 Aim for the Work

The overall objective of this work is to contribute to the automation of wear analyses for cutting tools. This by developing a method that can be used to automatically locate, classify, and measure wear on end mills. The focus will be on direct measurement methods and measuring the tool wear in a controlled environment outside the machine. The goal is to develop a process that can be applied in the industry. Thus, it is important that the method is accurate and easy to use. The focus will be on finding answers to the following three questions:

1. What is a suitable way of collecting image data of end mills for wear detection?
2. What are suitable image analysis methods for detecting wear on end mills?
3. How good are the methods found in questions (1) and (2) at measuring and classifying wear on end mills?

No strict performing goals are set since the exact requirements to deploy such a solution in an industrial environment is unknown. Instead, the focus will be to further expand the knowledge in the area of automatic tool wear measurement on end mills. With the aim of finding methods that can be evaluated and tested in an industrial setting.

1.3 Scope and Limitations

Many different approaches could be considered for automatic wear analysis. Since the focus of the work is to develop a system for wear analysis that can be deployed in an industrial environment, limitations will be put on the work to keep this focus. Further, the limitations will narrow down the scope of the work such that it can be finished within the time plan. The following limitations are put into place.

Developing a solution for three end mills: There are many types of cuttings tools [1]. To limit the time required for collecting data and training models the work will be limited to looking at three different tools. The type of tool will be limited to end mills, to ensure all tools tested have a similar geometry. Thus, simplifying the data acquisition process. The result will still be viable since end mills and other cutting tools share many common features.

Data acquired will be limited to 2D images: Different measuring techniques can be used for measuring wear. Examples are stereo imaging as used by [9] or focus variation microscopy that was used by [10]. Both create a 3D representation of the wear area. However most of the current literature regarding automatic tool wear measurements uses 2D images, examples are [11, 12, 13, 14]. The 2D representation of the wear area is also the most commonly used for determining tool life [1]. For these reasons, the work will be limited to a 2D representation of the wear area.

Only look at tool circumference: One end mill contains multiple cutting edges [1]. To capture all edges multiple camera angles are required. This will add complexity to the measurement setup. To limit the scope, and to put emphasis on the detection algorithms the work will be limited to only looking at the tool circumference. Thus wear on the tool end face will not be considered. This will allow a camera to capture all necessary angles by simply rotating the tool. This could be problematic in an application where the tool end face suffers from high wear. But all end mills will suffer from wear on the tool circumference, thus this is deemed to be a good starting point.

Not capturing the entire length of the end mills The entire end mill will not fit in one image. How much of the end mill will fit in a single image is determined by several parameters of the measurement setup that are not known in advance. To keep the focus on accurately measuring and classifying tool wear, the wear area analyzed by the algorithm will be limited to what fits in one separate picture. This should not have a major impact on how the result can be applied to entire end mills. The proposed methods will be scalable to different input sizes, thus allowing bigger images to be used in the future.

1.4 Thesis Structure

The structure of this thesis is as follows: Directly after the introduction is a theory chapter. It is split into four main sections: cutting tools, camera, optics and illumination, image analysis, and deep learning. The cutting tool section describes the geometry of an end mill, how it changes with different types of wear, and what methods are used to measure tool wear. The section camera, optics, and illumination focus on just those three areas, how they interact, and important parameters that control a final image. The two final sections look at image analysis tasks and how they can be solved with deep learning.

In Ch. 3 the methodology applied in order to design and implement a solution for automatic end mill wear measurement is described. Focusing on which tools were used for experiments and the design of hardware and software. Ch. 4 moves on from the methodology and describes the implementation process. This is split into two main areas, image acquisition, and image analyses. That describes how images of the end mills were captured, how the wear was identified in the images, and how the wear was quantified. The result and programs of the implemented methods are then described and discussed in Ch. 5. The result is flowed by a conclusion in Ch. 6 and lastly a discussion on possibilities for further work in Ch. 7.

2
Theory

2.1 Cutting Tools

Cutting tools remove material to form a final part. There exist several different methods for doing this and many different types of cutting tools. This chapter will focus on end mills and how they wear off.

2.1.1 End Mills

End mills are a type of cutting tool used in subtractive manufacturing processes. There exist many end mill types with different purposes. All from roughing end mills for removing material fast to ball end mills when milling contoured surfaces and specialty end mills for creating fillets, undercutting, and more [15]. Further, for each different type, there are many design choices, such as the number of flutes, cutter diameter, different materials, and surface coatings that are suited for different applications [1, p. 281-300]. What distinguishes end mills from other cutting tools is that they are designed to cut in the radial direction. This can be compared with drill bits that instead are designed to cut in the axial direction.

Fig. 2.1 shows the main features of an end mill as described by [16]. The area marked as primary and secondary relief surface is also called flank face. The cutting edge of the tool is where the rake face and primary relief surface meet.

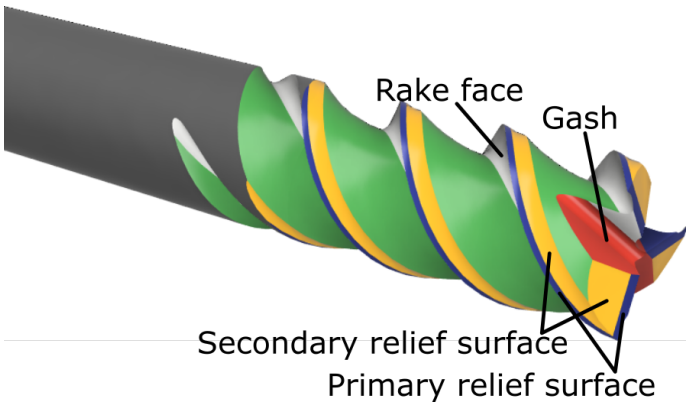


Figure 2.1: End mill surfaces. The white area is the rake face, blue is the primary relief surface, yellow is the primary relief surface and red is the gash.

2.1.2 Tool Wear and its Causes

Understanding how tool wear is important for maximizing the life of a tool. Tool wear also has an impact on the final quality of a produced part, thus it is important to know how it progresses during a machining operation to make sure parts are not damaged by worn tools.
There are several reasons why tools wear off. The most significant reasons are adhesive wear and abrasive wear [1, p. 538]. Abrasive wear is when hard particles abrade the surface of the tool. Abrasive wear is generally what determines the tool life, especially at lower cutting speeds. Adhesive wears occur when chips adhere to parts of the tool. When this happens there are two possibilities: 1) either the chip breaks away from the tool, taking small parts of the tool with it; 2) The other option is that parts of the chip stay attached to the tool, which leads to a build-up of workpiece material on the tool.

As temperatures get higher, adhesive wear and abrasive wear increase due to the tool softening, but other wear mechanics also become more significant. A softer tool increases the risk of plastic deformation. This is when the tool edge no longer can support the cutting forces and permanently deforms, thus changing its dimensions.

At high temperatures, there are other wear mechanisms like diffusion, oxidation, and chemical wear that will take over as the main causes of tool wear. Diffusion occurs when material in the tool forms a solid solution with chip material. This changes the material properties of the tool and weakens the tool surface, resulting in wear. Oxidation is when materials in the tool react with oxygen. It is most common at the surface of a part. Chemical wear occurs when the tool has a chemical reaction with either the workpiece or cutting fluid. It leaves a smooth wear area that can be hard to distinguish from marks left by grinding the tool. The reaction also changes the material at the tool surface, often resulting in a softer surface layer, thus, increasing other types of wear.

2.1.3 Wear Types

[1, p. 529-569] classify tool wear into 11 principle wear types. These are flank wear, crater wear, notch wear, nose radius wear, thermal cracking, mechanical cracking, edge buildup wear, plastic deformation, chip hammering, edge chipping or frittering, and fracture.

Flank wear is wear that occurs on the relief surface, it occurs in almost all machining operations. Fig. 2.2 shows an example of flank wear. Flank wear is generally a result of abrasion on the cutting edge, but can also be caused by chemical wear. Abrasion creates heat as well as damages the surface, the increased heat will lead to worse dimensional accuracy as well as a higher wear rate. Flank wear progression changes over cutting time, a new tool takes some time to wear in, and during this period the flank wear rate is high. Afterward, the wear rate decreases, resulting in the wear area steadily increasing in size until some critical point when the wear rate quickly increases until catastrophic failure of the tool.

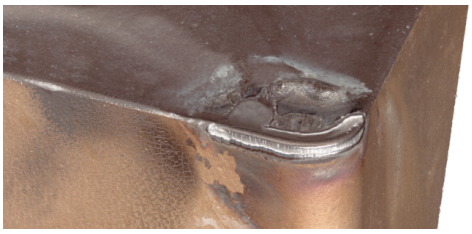


Figure 2.2: Flank wear. (Image from [17])

Crater wear leads to wear craters on the rake face of the tool [18], as shown in Fig. 2.3. In general, a limited amount of crater wear does not affect tool life, but excessive crater wear can lead to deformation or fracture of the tool, due to a weakened edge. Weakening of the cutting edge can also prevent the tool from being resharpened, thus limiting the tool's lifespan. Severe crater wear is most often a result of diffusion or chemical wear, but it can also be a result of abrasive wear.

Figure 2.3: Crater wear. (Image from [17])

Notch wear is localized wear that can occur on both the rake face and the relief surface [18]. It is most common at the depth of cut line where the tool is in contact with the shoulder of the workpiece. Notch wear at the depth of cut line is most often caused by abrasive wear, but it can also be caused by oxidation or by a chemical reaction. Similar to crater wear severe notch wear can prevent a tool from being resharpened or even lead to fractures.

Figure 2.4: Notch wear. (Image from [17])

Nose radius wear is wear on the trailing edge near the end of the relief face. It is due to abrasion, corrosion, or oxidation. Nose radius wear negatively impacts the surface finish of the final part.

Thermal cracking and *Mechanical cracking* create cracks that are either parallel or perpendicular to the cutting edge. Cracks perpendicular to the cutting edge, as in Fig. 2.5, are often a result of cyclic thermal loads whereas parallel cracks usually occur due to cyclic mechanical loads. Both types of cracking will increase the risk for tool fractures or chipping.

Figure 2.5: Thermal cracking. (Image from [17])

Build-Up Edge is a phenomenon where metal strongly adheres to the tool. It is often unstable and leads to a worse surface finish and tool chipping. Since the material is built upon the tool edge it also changes the depth of cut, resulting in a dimensional inaccurate tool. Fig. 2.6 shows how material can adhere to the tool.

Figure 2.6: Build-Up Edge. (Image from [17])

Plastic deformation of the cutting edge occurs when the tool can not support the pressure put on the edge, the result is deformation of the tool as shown in Fig. 2.7. Tool deformation most often occurs when using a high feed rate or a high cutting speed. This is since the hardness of the tool is reduced with higher temperatures. Plastic deformation leads to increased flank wear, a decrease in dimensional accuracy, and surface finish. It can also result in tool breakage.

(a) Edge depression. (Image from [17]) **(b)** Edge impression. (Image from [17])

Figure 2.7: Two versions of plastic deformation.

Chip hammering is when a tough or abrasive chip strikes the tool away from the cutting edge. It can lead to chipping or pitting or in the worst case a tool fracture.

Edge chipping or frittering results in a poor surface finish and increased flank wear. It may also lead to tool breakage. It occurs when the tool is unable to withstand cutting forces, thus parts of the edge are mechanically removed from the tool, as shown in Fig. 2.8. There are multiple factors that can cause edge chipping or frittering, these are: brittle tools, if the workpiece contains hard particles, or excessive vibrations.

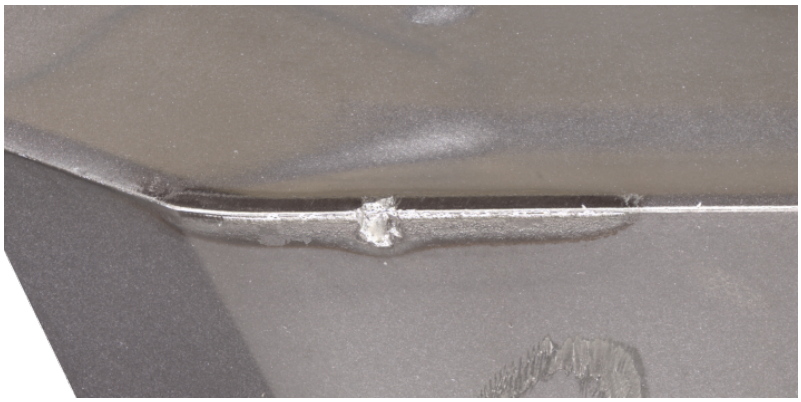


Figure 2.8: Chipping. (Image from [17])

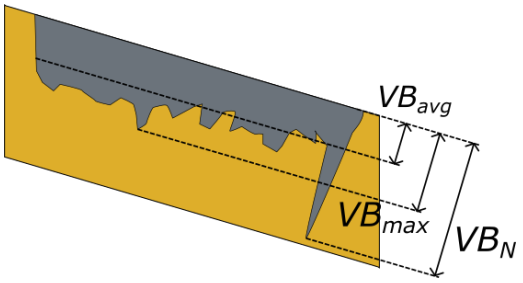
Tool fracture or breakage results in an unusable tool. Similar to chipping, it occurs when the tool is unable to withstand the cutting force. The difference from chipping is that the overload leads to a substantial portion of the tool breaking of, as shown in Fig. 2.9.

Figure 2.9: Fracture. (Image from [17])

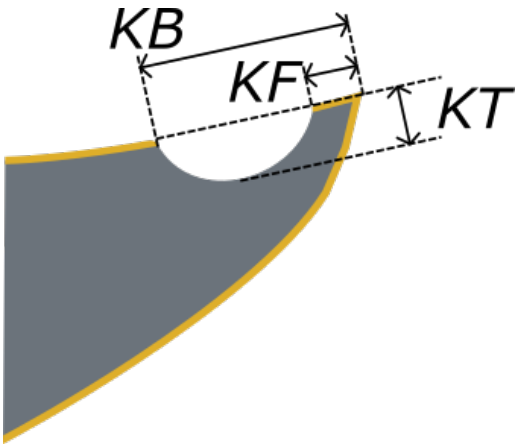
2.1.4 Wear Measurement

According to [1, p. 537] flank wear and crater wear are the most commonly measured forms of tool wear. Flank wear is normally classified by two measurements, these are the maximum and average width of the land wear area. If there is significant notch wear the notch is measured separately and the maximum of the central section of the wear land [1, p. 537], as shown in Fig. 2.10a.

For crater wear the most common parameters are the crater width, the crater land width, and the crater depth [1, p. 537], as shown in Fig. 2.10b.



(a) VB_{avg} is the average width of the flank wear, VB_{max} is the maximum width of the flank wear, and VB_N is the width of the notch.



(b) KB is the crater width, KF is the crater land width and KT is the crater depth

Figure 2.10: Parameters used when measuring flank wear, notch wear, and crater wear.

According to [1, p. 537], the most common way of collecting the measurements is with microscopes. More recently, general-purpose digital microscopes have been the industry standard. Microscopes only give a 2D representation of the wear, there also exist other methods that generate a 3D representation of the wear area.

2.2 Camera, Optics, and Illumination

This section presents important theory for the process of capturing images. This includes different options and settings to consider and how they affect the final image.

2.2.1 Camera Sensor

The camera sensor registers light and converts it into electrical signals. This is done by using light-sensitive elements (pixels) to record the number of photons that reaches the pixel and store it as a charge [19].

Sensors come in many different shapes, that can be sorted into three main categories: area sensors, line-scan sensors, and point sensors [20, p. 205]. Area sensors contain a grid of pixels, by exposing the pixels to light a two-dimensional image can be formed. Line-scan sensors are instead composed of a single line of pixels, thus it only captures a single one-pixel wide line, as illustrated by Fig. 2.11. A two-dimensional image can not be formed by simply exposing the pixels to light. Instead, an image is

formed by moving the subject relative to the camera and sequentially assembling the captured lines. A point sensor takes this further by only capturing a single pixel of information. Again this requires moving the subject relative to the camera to form an image.

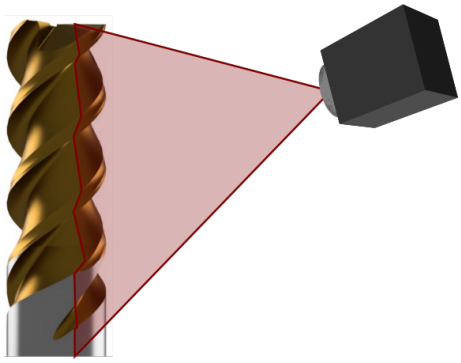


Figure 2.11: A line scan camera that captures a one pixel wide line along an end mill.

2.2.2 Exposure and Exposure Time

Exposure is the total amount of light energy that reaches a light-sensitive material and it depends on four factors: the light coming from the scene, how the light is transmitted through the optical system, for how long the sensor is exposed to the light, and the sensitivity of the image sensor to the light [21]. There are two main controls that affect the exposure, these are exposure time and aperture [21]. Aperture controls the exposure by controlling how much light is let through the optical system [22], whereas exposure time determines for how long the image sensor is exposed to the light. Further, how the scene is lit can be changed to affect the exposure of an image, if more light reaches the optical system it will increase the exposure.

What exposure to use depends on the information that is being collected. A camera sensor's dynamic range limits what information it can collect. The dynamic range is the ratio between the largest and smallest light intensity that the sensor can detect [21]. If the amount of light reaching the sensor is outside the sensor's dynamic range information will be lost. Depending on the lighting of a scene losing information due to the dynamic range may be unavoidable. Exposure controls what parts of a scene are inside the dynamic range. High exposure will give good details in darker parts of the scene, whereas lower exposure leaves more details in brighter areas.

There may be several ways of achieving any given exposure. However, when controlling the exposure there are other factors to take into account as well. The aperture will affect how much of a scene is in focus, see Sec. 2.2.4.2. For stationary scenes, exposure time only affects the exposure, however, if a scene is moving relative to the camera a longer exposure time will introduce motion blur [21]. Thus, there is a trade-off between the two options.

2.2.3 Sensor Gain and Bias

Gain control can be used to supplement the exposure settings. Where exposure is the amount of light that hits the sensor, gain controls the sensor output for a given exposure. Gain works by amplifying the sensor output signals [23, p. 273]. This can help in scenarios where the exposure controls are insufficient to achieve an adequate result. The drawback of using gain is that it also amplifies any noise.

It can also be possible to adjust gain as well as a bias for each individual pixel [24]. This is especially important for line scan cameras where many lines are used to form an image. If the pixels are not properly calibrated, lines will appear in the image, due to the repetition of the same pixels.

2.2.4 Optics

The optical system is one of the key parts of image acquisition. A lens is needed to collect light and form an image on the sensor.

2.2.4.1 Focal length and Magnification

Focal length is a property of a lens, that describes where parallel incident light will converge to a single point [22]. For non-parallel light rays, emanating from a single point will instead be focused at a distance v from the lens, where v can be described by:

$$\frac{1}{u} + \frac{1}{v} = \frac{1}{f},$$

f is the focal length, u is the distance from the first nodal plane to the object and v is the distance from the second nodal plane to the image plane, as seen in Fig. 2.12. The nodal planes are again properties of the lens.
The focal length combined with the size of the image sensor will determine the field of view [25]. Further, the field of view combined with the distance to the object gives the magnification of the system. The magnification M of a lens is defined as the ratio between the size of the actual object and the size of that object on the image plane. It can be calculated by the following equation:

$$M = \frac{I}{O} = \frac{v}{u} = \frac{f}{f-u},$$

where I and O are the size of the image and object respectively. f is the lens focal length, u and v are distance defined as in Fig. 2.12 [22].

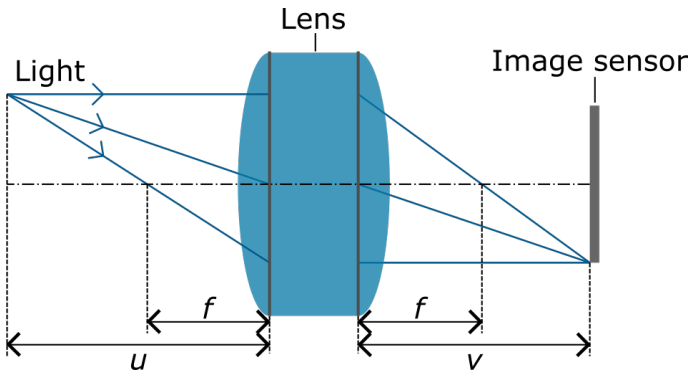


Figure 2.12: Model of a thick lens. f is the focal length, u is the object distance and v is the image distance. The grey lines inside the lens illustrates the two nodal planes.

2.2.4.2 Aperture and Depth of Field

Aperture is the light-transmitting ability of a lens [22]. It is the aperture combined with the exposure time that controls the amount of light that reaches the image sensor [26]. The aperture of a lens is most commonly measured in *relative aperture/f-number*, relative aperture is defined as:

$$N = \frac{f}{d},$$

where N is the relative aperture, f is the focal length and d is the effective clear diameter of a lens [26], as seen in Fig. 2.13.

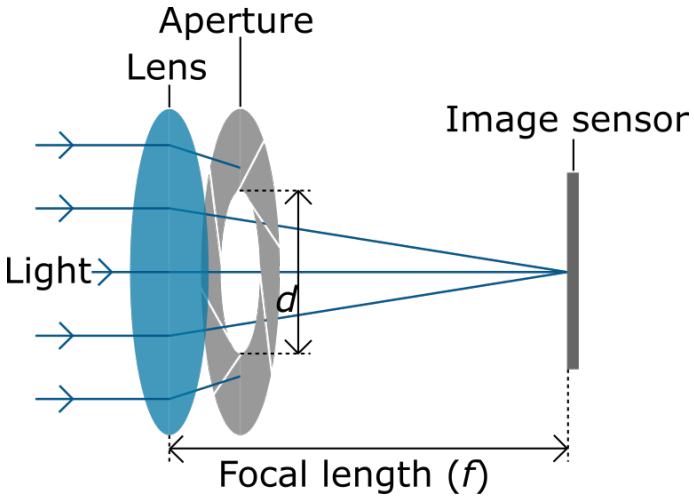


Figure 2.13: Aperture of an optical system.d is the effective clear diameter of a lens.

The definition of relative aperture assumes that the lens is focused to infinity. For closer objects, this approximation will not apply, instead, the *effective relative aperture/effective f-number* should be used. The effective relative aperture can be calculated as:

$$N' = N(1 + M),$$

where N' is the effective aperture, N is the relative aperture and M is the magnification [26].

The aperture also affects the depth of field. When the lens is focused at a close distance the depth of field can be approximated with eq. (2.1) [26]. Here the depth of field is defined as the distance between the farthest and nearest point that an object can be placed from the lens whilst keeping the diameter of the circle of confusion (CoC) smaller than C. The circle of confusion is a measurement of how well light from a single point on the object converges on the image sensor. More intuitively depth of field is the distance between the farthest and closest object that is in acceptable focus.

$$T = \frac{2CN'}{M^2}, \tag{2.1}$$

where T is the depth of field, C is the diameter of the CoC, N' is the effective aperture and M is the magnification, as seen in Fig. 2.14. There is no definition of what is an acceptable size for the CoC, an acceptable value must be determined for every separate scenario. One important factor in the choice of C is the sensor's pixel pitch. A sensor with a large pixel pitch can handle a larger CoC. This is since an image loses focus when a single point on the subject covers multiple pixels on the image sensor. A larger pixel pitch will limit the number of pixels inside the CoC.

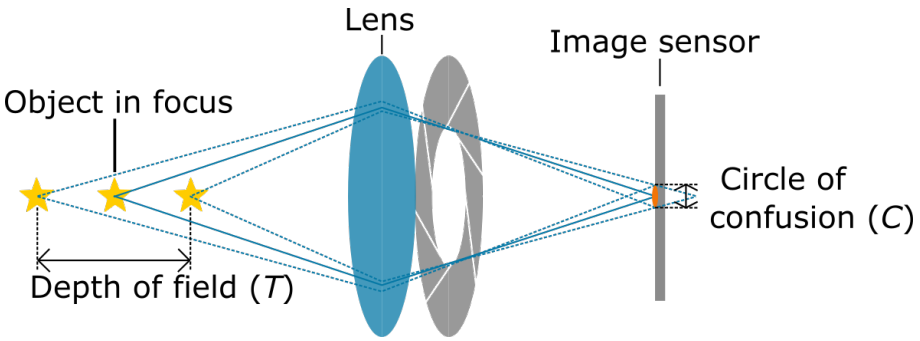


Figure 2.14: Depth of field of an optical system.

2.2.5 Illumination

Cameras record how light has interacted or not interacted with objects. Thus, illumination is one of the key aspects of a vision system. Poor illumination can lead to missing information in an acquired image, information that may be impossible to regain in an image analysis step [24, 27].

Light can interact with objects by absorption, reflection, scattering, or by being transmitted in the case of a transparent object. How this happens depends on both the properties of the light and the object [20, p. 66-76].

2.2.5.1 Color and Wavelength

The wavelength of light is one of the parameters that determine how light interacts with matter. Absorption, transmission, and scattering of light all depend on the wavelength. Absorption of some wavelengths while reflecting all other wavelengths is what gives a material its color [20, p. 66-76].

Thus, the wavelength of the light used to illuminate an object will affect how a camera sees it. This holds for both color cameras as well as monochromatic cameras. Absorbents will make objects and features with the same color as the light appear bright as they reflect the light. While objects of a different color will be darker as they absorb most of the light [24, p. 51].

2.2.5.2 Light Intensity

As described in Sec. 2.2.2 and 2.2.4.2 increasing exposure time or aperture to increase exposure both have negative side effects. A third option to increase the exposure is to increase the light intensity reaching the lens by adding more light to the scene. This can be done by simply adding more or more power full lights, however, there are other options. [24] presents two methods to increase effective light output: strobing lights and focused lighting. Strobing is achieved by turning on a light only during the camera's exposure time. Depending on the exposure time and how often a picture is taken this can allow significantly higher currents than if the light was continuously on. Between pulses, the light must have time to cool down. The second method is to use a lens to focus the light on a specific spot or line, thus increasing the light intensity in that area.

2.2.5.3 Illumination Geometry

One of the most important factors when it comes to illumination is the angle at which the light falls on the object. According to [24, 28] there are two major lighting options: *front lighting* and *backlighting*. When using front lighting, the light source is placed on the same side of the object as the camera, compared to backlighting where the light source is on the opposite side as the camera. [24, 28] further divides the different angles into bright field illumination, coaxial illumination, dark field illumination, and diffuse/dome illumination. The light angle relative to the camera for bright field illumination, coaxial illumination, and dark field illumination are shown in Fig. 2.15.

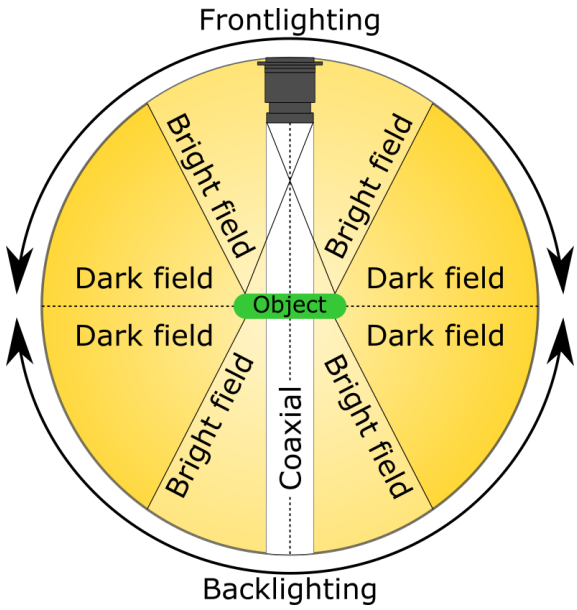


Figure 2.15: Different lighting angles in relation to camera position.

For bright field illumination, the light source has a direct path to the camera [20, p. 243-244]. The light reaches the camera sensor either by reflecting of the object as in the case of front lighting or by having the object between the light source and the camera in the case of backlighting. In this way, a large amount of light will reach the image sensor [20, p. 243-244]. When using bright field front lighting, flat surfaces that reflect the light to the imaging sensor will show up bright on the image, and non-flat features (e.g. scratches) will be darker as they reflect the light in many directions [29].

Coaxial illumination is a subset of bright field illumination, where the light is emitted in line with the camera. It can be realized by reflecting the light of a semi-transparent mirror [20, p. 243-244]. Coaxial illumination can also be collimated (all light rays are parallel). This will make any surface feature such as depression or bumps reflect the light outside the field of view, only surfaces parallel to the light, and the image sensor will reflect light back to the sensor [24].

Darkfield illumination does not directly target the light at the image sensor, instead only scattered light reaches the sensor [20, p. 244-245]. For a flat surface, this means that most of the light is reflected out of the view of the camera. Instead rough surfaces will show up bright in the image as they scatter the light in many directions [20, 24]. Darkfield backlight illumination can not be used on opaque objects since the light can not scatter in the direction of the camera [30].

Unlike the previous methods, diffuse illumination has no distinct direction. It illuminates the object equally from all directions. Thus, no shadows will be cast on the object, however, there will still be bright spots if the object geometry reflects the light directly to the image sensor [20, p. 243-244].

2.3 Image Analysis

Image analysis is a very wide field, this section introduces three concepts, image classification, object detection, and image segmentation.

Image classification is the process of associating images with labels, every image gets one label [31]. Thus, the entire content of the image has to be considered when classifying an image. For this reason, image classification typically refers to images where there is a single object being analyzed. In contrast, segmentation and object detection can find multiple objects in one image and associate them with separate labels [32].

2.3.1 Types of Segmentation and Object Detection

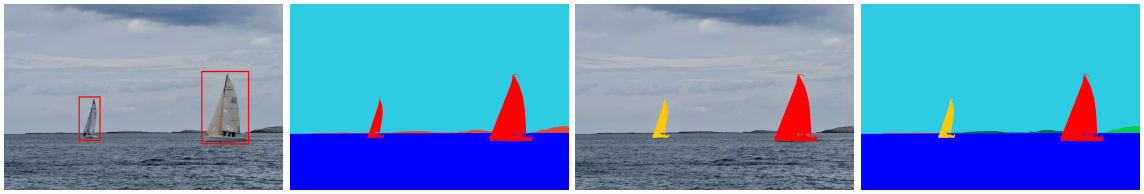
There are different ways of identifying objects in an image. [32] describes the following four methods: object detection, semantic segmentation, instance segmentation, and panoptic segmentation.

Object detection is the task of identifying, locating, and classifying instances of objects in an image. An example can be seen in Fig. 2.16a, where two sailing boats are detected.

Semantic segmentation is the process of assigning a class label to each pixel in an image. The labels are assigned based on the objects in the image. Fig. 2.16b shows an example of semantic segmentation where the sky, islands, and boats are detected.

Instance segmentation is a combination of objected detection semantic segmentation. It is the process of assigning a label to an individual pixel based on the object instances in the image. An example of this is seen in Fig. 2.16c, where the two boats get different labels, but it does label all pixels.

Panoptic segmentation combines instance segmentation and semantic segmentation. All pixels in an image are associated with a class label and an instance, as seen in Fig. 2.16d.



(a) Object detection **(b)** Semantic **(c)** Instance **(d)** Panoptic

Figure 2.16: Examples of different types of segmentation applied to the same image.

2.3.2 Evaluation Metrics

In binary classification tasks, a classifier's performance can be defined based on the number of True positives (TP), False negatives (FN), False positives (FP), and True negatives (TN) [33]. TP is how many of the positive instances have been classified as positive, FN is how many of the positive instances have been classified as negative, and so on. From this several metrics can be constructed, of which two are, precision (p), and recall (r), defined by eq. (2.2), and (2.3). Precision describes how many of the instances classified as positive actually are positive and recall measures how many of the positive instances are classified as true.

$$p = \frac{TP}{TP + FP} \tag{2.2}$$

$$r = \frac{TP}{TP + FN} \tag{2.3}$$

The same metrics used for image classification can be used for image segmentation. But they need to be combined with other metrics, to determine TP, FP, TN, and FN. The most populated metric used for image segmentation is the intersection over union (IoU) [34], also known as the Jaccard index [35]. The IoU gives a number from zero to one that describes how well the detected region overlaps with the intended region. This can then be directly used as an evaluation metric or with a threshold to determine TP, FP, and FN detection. IoU is defined as following:

$$\text{IoU} = \frac{|A \cap B|}{|A \cup B|} = \frac{\text{}}{\text{}}, \tag{2.4}$$

where A and B be are two sets. In the case of image segmentation, A is the set of all pixels in the ground truth and B is all the pixels in the prediction.

2.4 Deep Learning

Deep learning is a sub-field of machine learning, that deals with neural networks with many layers. This section presents some core concepts of deep learning, related to image segmentation and classification.

2.4.1 Artificial Neural Networks

Artificial neural network is an attempt to emulate how biological neurons work [36]. The basic building block of an artificial neural network is the artificial neuron. An artificial neuron consists of a set of inputs, a weight for each input, a bias term, an activation function, and finally an output, Fig. 2.17 shows a representation of a neuron. The output y from an artificial neuron is calculated accordingly:

$$y = f\left(b + \sum_i x_i w_i\right),$$

with the variables defined as in Fig. 2.17.

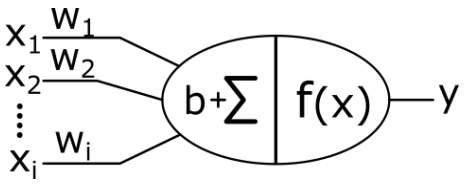


Figure 2.17: Representation of an artificial neuron with inputs $x_1, x_2, \ldots, x_i$, weights $w_1, w_2, \ldots, w_i$, bias b, activation function $f(x)$ and output y.

An artificial neural network is created by connecting the output of one neuron to the input of other neurons. There exist many different architectures for how the neurons are connected to each other. One of the most common is the fully connected neural network (FCNN) where the neurons are organized into layers, as in Fig. 2.18 [36]. There are three types of layers in an FCNN, these are: input, output, and hidden layers. As the name suggests input and output layers are used to input and extract data from the network. The neurons in the input layer do not have any weight or activation functions, they simply store the input value. The hidden layers are not accessible, instead, they allow the network to solve more complex tasks. For practical reasons the activation function used in hidden layers should be non-linear, otherwise, the network will simply evaluate to a linear function.

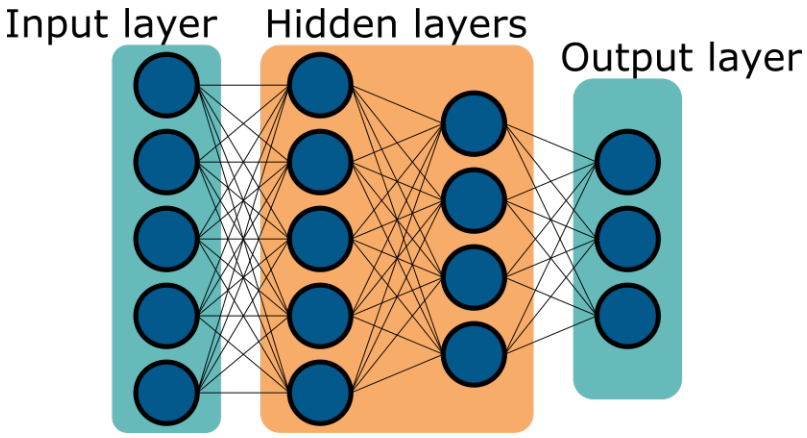


Figure 2.18: A schematic representation of a fully connected neural network with two hidden layers. Every neuron is represented with a circle.

Neural networks are trained by solving an optimization problem. This is done by defining a loss function, which is minimized by selecting appropriate weights and biases for each neuron in the network. Training the network is then done in two steps. The input defines the values in the input layer, the information is then propagated through all the hidden layers to the output layers where a loss is calculated [37]. This first step is called forward propagation. After the loss is calculated, the weights and biases of the network should be optimized to decrease the loss. Optimization of the parameterize is done with gradient descent [38]. Calculating the gradient of the loss function is done with back-propagation, this works by propagating the loss from the output layer to the input layer [37]. These two steps are repeated until a local minimum is found or the performance of the network is satisfactory.

2.4.2 Convolutional Neural Networks

Convolutional neural networks (CNN) are another architecture that builds on the FCNN. They are proven to perform very well in computer vision tasks, for example, image segmentation and image classification [39].

CNNs introduce two new layer types, convolution layers, and pooling layers. Fig. 2.19 shows a schematic representation of convolution layers and pooling layers in a convolutional neural network. The convolution layer extracts local features from the data, instead of global features like an FCNN. This is achieved by scanning a kernel over the input data, the results are stored in a feature map. A kernel is a matrix of weights that are multiplied with the input, by changing the weight the kernel can be used to detect the location of a specific feature in the input data. A kernel could for example be designed to detect horizontal edges in an image. The kernel is learned when training the network, thus there is no need to manually define the weights to extract relevant features. The following equation is used to calculate a feature map:

$$L_{i,j}^{(1)} = f\left(\sum_a \sum_b h_{a,b} \cdot L_{i+a,j+b}^{(0)}\right),$$

$h_{a,b}$ is a kernel, $L_{i,j}^{(0)}$ is the input data, $L_{i,j}^{(1)}$ is the resulting feature map and f is an activation function.

Pooling layers are used to down-sample and reduce the complexity of a network [39], they also reduce the problems with overfitting [40]. Pooling layers function similarly to convolutional layers, they apply a function to small regions of the feature map. The most common type of pooling layer is max pooling, where the most prominent feature is extracted from a region in the feature map. Other options are min pooling and average pooling.

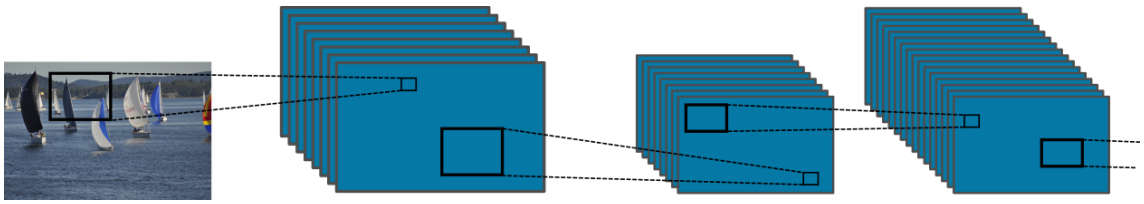


Figure 2.19: A schematic representation of a convolution layers and pooling layers in a convolutional neural network. One blue square represents a feature map.

Convolutional layers and pooling layers can be used in combination with fully connected layers. This is common for image classification networks, where a feature extractor is built using convolutional layers and pooling layers are followed by fully connected layers. Examples are AlexNet [41] and the ResNet [42] family of networks. If no fully connected layers are used in a network, it is called a fully convolutional network (FCN). FCN was introduced by [43] and they are often used for image segmentation.

2.4.3 Mask R-CNN

Mask R-CNN is a framework designed for instance segmentation. It was introduced by [44] and is an extension of the Fast R-CNN framework. The framework has a two-stage architecture. The first stage generates proposals that are sent to the second stage. A proposal is a region that likely includes a region of interest (ROI). The second stage refines the proposal and generates a bounding box, a mask, and classifies the object in the bounding box. Fig. 2.20 shows the components that make up the two stages. The components are the backbone, region proposal network (RPN), ROI pooling and ROI Classifier, Bounding Box Regressor, and Mask generator.

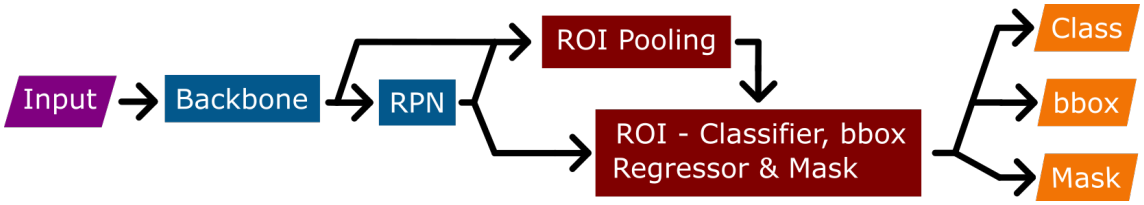


Figure 2.20: Schematic representation of the Mask R-CNN framework. Blue boxes are the first stage, red boxes are the second stage, the purple box is the input and the orange boxes are the outputs.

The backbone is an FCN network that serves as a feature extractor. The region proposal network takes the feature map and generates region proposals. The RPN consists of a neural network that scans over regions in the feature map. Each region that is scanned is called an anchor. The RPN classifies every anchor as either foreground or background. If an anchor is classified as for foreground, the RPN refines the bounding box and passes it on to the next stage. Anchors that are classified as background are thrown away.

The second stage classifies, further refines the bounding box, and generates a mask for each ROI that is proposed by the RPN. These are the final output of the Mask R-CNN framework. Unlike the RPN there can be more than two classes at this stage, but there is always a background class, that discards the region. CNNs are used to generate all the bounding boxes, masks, and labels. CNN's are not good at handling different-sized inputs. This is a problem since the ROI proposed by the RPN is not of a fixed size. This is fixed by the final component, ROI Pooling, which is used to create a fixed input size for the CNN used in the final stage. An illustration of these steps can be seen in Fig. 2.21.

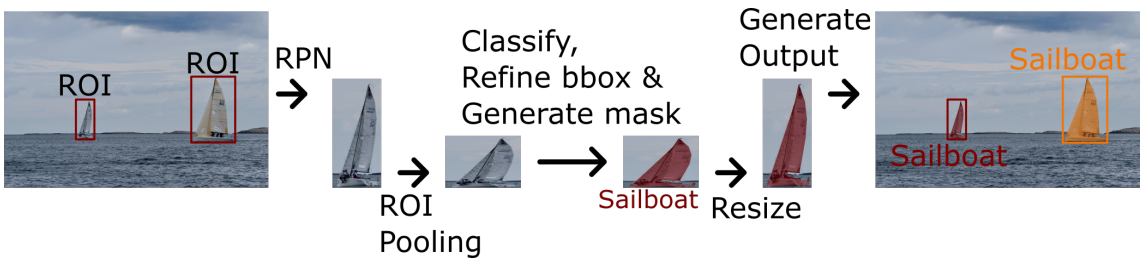


Figure 2.21: An illustration of how the components of the Mask R-CNN framework interact to generate a output. It starts by using the RPN that identifies ROI. The ROIs are then converted to a fixed size. A mask and classification is generated and the bonding box is refined. Lastly the generated data reintroduced to the image coordinate system.

3

Methodology

The core concept is to do offline wear measurements at the end of the tool life cycle. This was achieved by taking images of the end mills after they are taken out of the machine, then relevant wear metrics were extracted from the images. Fig. 3.1 shows how this fits into the tool life cycle, from tool storage to machining, data collection and finally storing the data for further analysis.

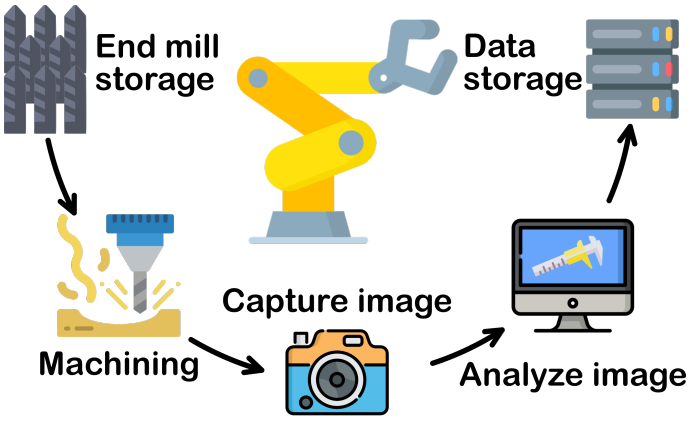


Figure 3.1: How the data collection and image analysis are integrated into the tool life cycle. From storage and machining to capturing an image, analyzing the image, and storing the data for later use. (Images from [45, 46])

3.1 Tools

Three tool variants were chosen as test subjects. For reference, these tools will be referred to as *T8*, *T10*, and *T12*, with the number referring to the diameter of the tools. The T8 tool can be seen in Fig. 3.2a, T10 in Fig. 3.2b, and T12 is shown in Fig. 3.2c. All tools have been used to machine Haynes 282 which is a titanium super alloy and are taken at the end of their usable tool life. The three tools were selected to encompass some of the varieties that exist between end mills. T8 is an 8 mm, six flute end mill with titanium carbonitride coating, T10 is a 10 mm, six flute end mill with a titanium nitride coating and T12 is a 12 mm end mill, four flute end mill with a titanium nitride coating.

(a) T8, 8 mm end mill, six flute end mill with a titanium carbonitride coating

(b) T10, 10 mm end mill, six flute end mill with a titanium nitride coating

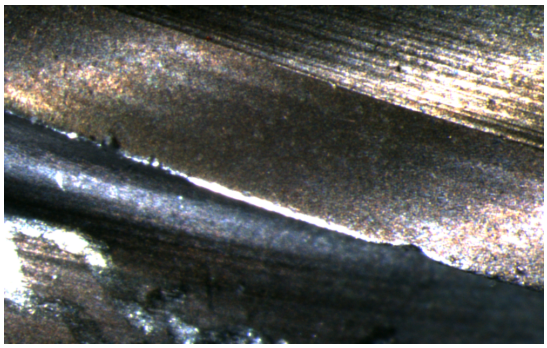
(c) T12, 12 mm end mill, four flute end mill with a titanium nitride coating

Figure 3.2: The three end mills that were chosen as test subjects

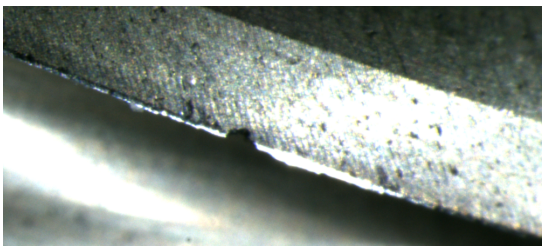
3.2 Investigated Wear Types

As described in Sec. 2.1.3, there exist many different types of cutting tool wear. All wear types were not targeted in this work. The focus was on detecting flank wear, chipping, and fractures. Flank wear was chosen since it is commonly used by standards for determining tool life [1, p. 548], it is for example recommended by the *ISO 8688-2* standard [47]. Chipping and fractures were also added to the list since they were a common problem for the tools analyzed in this thesis. Especially chipping and fractures at the tip of the tools were a big problem and this was generally what determined the tool life. Fig. 3.3 shows an example of wear on some of the end mills. Due to their similarities chipping and fractures will be treated as the same from here on and simply referred to as fractures.

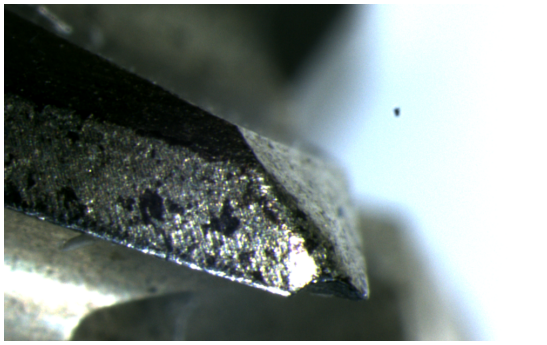
The average depth of the wear area VB_{avg} and the maximum depth of the wear area VB_{max} were measured for flank wear. For the tools selected, the flank wear was approximately 30 µm - 120 µm. The fractures were quantified by their maximum depth, denoted by FW_{max}. The fracture depth was in most cases similar to the flank wear depth, but some outliers exist with a depth up to 400 µm.



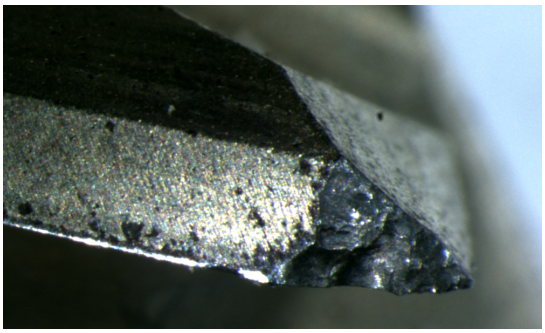
(a) Two fractures and flank wear on a T8 tool.



(b) Flank wear and a fracture on a T10 tool.



(c) Tip fracture and flank wear on a T10 tool.



(d) Tip fracture and flank wear on a T12 tool.

Figure 3.3: Wear examples on the selected end mills. Flank wear is the shine area along the edge and fractures are the missing pieces of the edge or tip.

3.3 Image Acquisition

The first step in measuring the wear was to acquire images of the end mills. This requires a measurement setup that makes it possible to capture images of the end mills. Since this thesis aims to create a system for automatic tool wear measurement, this setup should allow for repeatable and automatic image acquisition. It also needs to be adaptable to different tool geometries. To accomplish this the measurement setup consisted of a camera, a lens, and lighting for taking the pictures. Further, the measurement setup needs a motion system that makes it possible to move the end mill relative to the camera. This to allow the camera to capture images around the circumference of the tool and makes it possible to use the same setup for different diameter tools.

To capture images a line scan camera was used, this since a line scan camera is able to capture flattened images of the end mills without distortion [48]. The camera needs to be paired with a suitable lens. When selecting the lens the main focus was on getting adequate magnification to detect and measure the wear. The working distance of the lens was also an important factor, a short working distance would make it hard to install lights for illumination and it could possibly hinder the place-

ment of end mills. Illumination was also experimented with by looking at how the lighting principles discussed in the Sec. 2.2.5 affect the image.

The camera was mounted on linear rails that allow movement in three axes. This allows for fine-tuning of the camera position and adapting the setup to different diameter end mills. A rotation table was used to hold the end mill. An unwrapped image of an end mill's circumference was created by rotating the end mills at a constant speed and simultaneously scanning multiple lines with the camera.

For machine learning, a large and diverse data set is vital to prevent overfilling of a model [49]. The number of images acquired was limited due to time constraints. Even with a fully automatic setup for capturing images, manual annotation was required for training. To increase the data set data augmentation techniques were used. As suggested by [50], the augmentation methods were selected to simulate possible variations in the image capturing process.

3.4 Image Analysis

The image analysis consisted of two steps: segmentation and quantification. First, an image segmentation algorithm was used to locate and classify the wear in the images. After the segmentation, a quantification algorithm was applied to the segmentation result, to extract key metrics. That being VB_{avg} and VB_{max} for flank wear, as defined in Sec. 2.1.4 and the maximum fracture width.

Several different segmentation methods have been used in previous literature [11, 12, 13, 14, 51]. These can be split into two groups: conventional computer vision methods and machine learning methods. Machine learning with convolutional neural networks has shown great results for image segmentation tasks over the last years [32], further both [12] and [11] show promising results using machine learning for wear detection. Thus, the machine learning approach was used in this thesis, utilizing CNNs to segment the wear areas. The machine learning algorithm was complemented with conventional computer vision to extract quantifiable measurements from the segmented areas.

VB_{avg}, VB_{max} and, FW_{max} are standard parameters used to quantify wear. These were the main parameters used for evaluating performance. The performance of the image segmentation algorithm was also evaluated. For this IoU was the main metric used since it is the most popular metric used for image segmentation benchmarks [34]. Precision and recall were also used to evaluate the image segmentation step.

Implementation of all algorithms was done with Python 3.7 using the package presented in Tab. 3.1. For more information about the python environment, see appendix B. For training neural networks a laptop equipped with an NVIDIA RTX A5000 laptop GPU with 16 GB of GDDR6 memory was used.

Table 3.1: Python package used.

Package	Version	Task
pytorch	1.11.0	Machine learning framework
torchvision	0.12.0	Datasets, model architectures, and image transformations for computer vision
numpy	1.21.6	Math library
scipy	1.7.3	Scientific computing and technical computing
Pillow	9.1.0	Opening, manipulating, and saving images
cython	0.29.28	C-Extensions for Python
matplotlib	3.5.1	Plotting library
scikit-image	0.19.2	Image processing
opencv-python	4.5.5.64	Image processing and computer vision
pycocotools-windows	2.0.0.2	Assists in loading, parsing and visualizing the COCO dataset
tensorboard	2.8.0	Visualization and tooling for machine learning experimentation

step

4

Implementation

This chapter explains how the methods from the methodology chapter are implemented.

4.1 Image Acquisition

This section deals with the design of the image acquisition setup, and how it is calibrated and used to capture images. Finally, it describes the acquired data set used to train and evaluate the algorithms.

4.1.1 Image Acquisition Setup

The image acquisition setup can be seen in Fig. 4.1 and 4.2. The main parts are a camera, a lens, lights, and a motion system. End mills are held in a tool holder, the tool holder is then placed in the center of a rotating table.

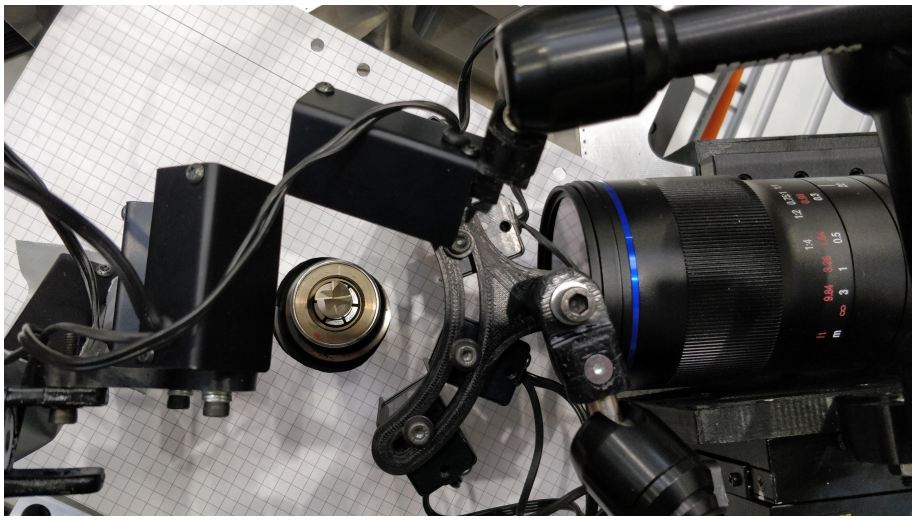


Figure 4.1: Image acquisition setup seen from above. The end mill and tool holder is seen in the center surrounded by lights and a line scan camera. The camera can not be seen in the image, only the lens is visible in the right side of the image.

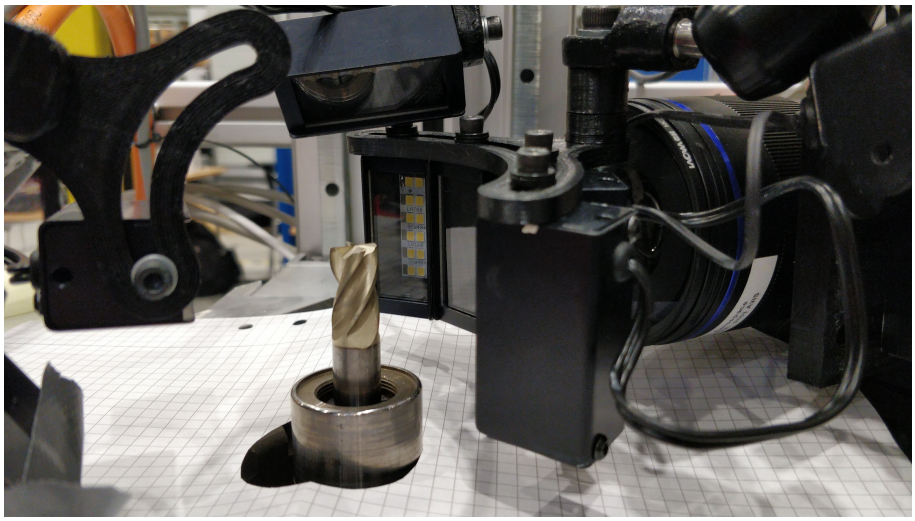


Figure 4.2: Image acquisition setup seen from the side. The end mill and tool holder is seen in the center surrounded by lights and a line scan camera. The camera can not be seen in the image, only the lens is visible in the right side of the image.

4.1.1.1 Camera

The camera used was the *Basler racer raL2048-48gm*. Specification for the camera can be seen in Tab. 4.1.

When all pixels on a sensor are exposed to uniformed light all pixels should have the same output value. This is not the case due to variations in performance between the pixels. To ensure a uniform output value from all the pixels, each pixel was calibrated according to Sec. 10.9 in the user manual [52, p. 184-190]. The calibration applies a unique offset and gains to each pixel.

Table 4.1: Specifications for Basler racer raL2048-48gm [52].

Description	**Value**
Resolution	2048 pixels
Effective sensor diagonal	14.34 mm
Pixel Size	7 µm x 7 µm
Color/Mono	Mono
ADC bit depth	12 bit
Line rate	100 Hz - 51 kHz
Lens mount	c-mount
Interface	Gigabit Ethernet
Software	Basler pylon Camera Software Suite

4.1.1.2 Lens

The lens used was a *Laowa 100mm f/2.8 2x Ultra Macro APO*. Specifications for the lens can be seen in Tab. 4.2. It is made for a full-frame sensor (24 × 36 mm), thus it works for the 14.34 mm effective sensor diagonal of the camera. A c-mount adapter was used to mount it to the camera.

It was chosen due to the stated 2x magnification at a focusing distance of 247 mm. This gives a theoretic pixel resolution of 3.5 µm, which is adequate for detecting the wear, while the focusing distance leaves room for lights between the lens and the end mill. Further, the lens has a variable aperture that makes it possible to change the depth of field and how much light reaches the camera sensor.

Table 4.2: Specifics for Laowa 100mm f/2.8 2x Ultra Macro APO [53].

Description	Value
Resolution	2048 pixels
Focal length	100 mm
Max. Apature	f/2.8
Format compatibility	24 × 36 mm (Full Frame)
Min. Focusing Distance	247 mm
Max. Magnification	2:1

4.1.1.3 Illumination

For illumination, six of Latab's 50 mm line lights were used. The specific lights used were the white quad line light (SAW4 4050) and the red quad line light (SAH4 4050) [54], only two of the four red lights were used. Fig. 4.3 shows the lighting geometry. Most of the illumination came from the white lights. These were set up in a semi-circle around the end mill to provide even illumination around the end mill. The two white lights closest to the camera axis mostly illuminated the primary and secondary relief surfaces. The two outer white lights were used to provide light into the flutes, thus getting a better contrast between the cutting edge and the background. Relative to the relief angle the two outer lights were set up as dark field lighting, this resulted in shine on the worn parts of the edge as explained in Sec. 2.2.5.3. Especially the position and angle of the right-most light (looking from the camera's perspective) had a big impact on the reflected light from the worn area.

The two red lights were positioned to enhance the contrast between the background and the tip of the tool. One light was used as a bright field backlight to enhance the outline of the tool tip. The other red light was positioned on the right side of the camera axis, above the white lights, at a downward angle. This provides illumination into the gash.

All lights were continually powered with a constant current. The white lights were operated at their maximum rated current of 1 A, to provide maximum illumination. A lower current of 300 mA was used for the red lights. This was to prevent glare that was outside the dynamic range of the camera.

White light was used as the main source of illumination since the color of the end mills differs depending on their coating or lack of a coating. Due to the reflection and absorption phenomenons explained in Sec. 2.2.5.1 white light should perform uniformly over different coatings. Red lights were used for all other angles since no more white lights were available.

(a) Light positions seen from above.

(b) Light positions seen from the side.

Figure 4.3: Light positions used in the image acquisition setup. The yellow lights represents the white lights.

4.1.1.4 Motion system

The motion system had three tasks, most importantly it allows for photographs to be taken of the entire end mill, it also makes it possible to adapt the setups to different diameter and length end mills. Finally, it allows for fine-tuning of the camera position. Making it possible to center the line scan camera on the end mill, and adjust the distance to the end mill for a sharp focus.

The motion system consisted of two motorized axes and two manual axes. The motorized axes are for rotating the end mill and moving the camera up and down along the end mill (z-axis). These are used during the image capture process (see Sec. 4.1.2), thus they need to be motorized to capture images at a reasonable rate.

For rotation, the *EMME-AS-100-M-HS-AMB* Festo servo [55] was used in combination with a 20:1 gear ratio. The end mill was then mounted to the output of the gearbox. The servo has a typical angular accuracy of 20 arcmin, thus in theory the end mill should be positioned with an accuracy of 1 arcmin. The linear z-axis was driven with a ball screw controlled by an *EMMT-AS-60-M-LS-RMB* Festo servo [56]. The Festo servo drivers have an Ethernet interface allowing for easy integration with the camera.

The manual axes adjusted the camera distance to the end mill and the lateral position of the camera. This was used for fine-tuning the image. The distance control was also used when changing between end mills of different diameters. This made it possible to analyze different diameter tools without changing the focus and recalibrating the pixel resolution. The manual axes were constructed using two Melles Griot 65 mm linear translation stages.

4.1.2 Image Capture Process

An image was captured by rotating an end mill at a constant speed and simultaneously scanning lines with the camera. Fig. 2.11 illustrates how one line is captured. By scanning multiple lines while rotating the end mill a flattened two-dimensional image of the end mill can be constructed. Due to the length of the end mills, one rotation was not enough to capture the full length of the tool, thus several rotations were used with different camera positions. Between every picture, the camera was moved 7 mm along the z-axis. Festo Automation suite was used to control the servos and Basler pylon Camera Software Suite was used for controlling the camera. Using Pylon it was possible to save each rotation as a separate two-dimensional image, no stitching of the individual lines where required.

When capturing images this way the rotational speed of the end mill must match the line rate of the camera. Given an end mill and a line rate, the rotational speed of the end mill in degrees per second is given by:

$$\Omega = 360\frac{\nu_l}{N}, \tag{4.1}$$

$$N = \frac{D\pi}{k}, \tag{4.2}$$

where ν_l is the line rate, N is the number of line scans needed to capture one rotation of the tool, D is the tool diameter, k is the pixel resolution and Ω is the angular velocity.

4.1.3 Calibration of Pixel Resolution

In a line scan of a stationary end mill, there is no good information to use for calibration. Thus, it is not possible to calibrate the pixel resolution using the normal image setup as done by [14]. Instead similarly to [13], a separate object was placed in front of the camera for calibration purposes. This was done by using a glass ruler with a 0.1 mm scale in place of the end mill. For lighting, the same backlight as in the regulated setup was used.

The pixel resolution was then calculated using eq. (4.3), giving a resolution of 3.6 µm/px and a magnification of 1.94x. 3.6 µm/px is deemed to be an acceptable resolution considering the size of the wear. With the camera's resolution of 2024 px, the height of each scan will be 7.4 mm.

$$k = \frac{\ell}{N}, \tag{4.3}$$

where k is the pixel resolution, ℓ is the distance between two points on the object and N is the number of pixels between the same points in the image.

It is crucial that the ruler is at the same distance from the camera as the end mill. Otherwise, the calibration will be wrong due to the magnification changing with distance, as described in Sec. 2.2.4.1. To prevent this the ruler was attached to the same tool holder as the end mills. For verification, the pixel resolution was also measured at different distances within the depth of field. There was no measurable difference, thus as long as the end mill is in focus the calibration should be accurate.

This only allowed for calibration in one direction. However, since the pixels are square and with the assumption that the angular speed Ω and line rate ν_l follow Eq. (4.1), the pixel resolution will be the same in both directions. Differences in the internal clocks of the servo motor and the camera would result in Ω and ν_l having the wrong relationship. To confirm the pixels were square, the number of pixels in one rotation was counted. If Ω and ν_l have the correct relation the number of pixels for one rotation should be as defined by eq. (4.2). Measurements were done by capturing an image of more than a full revolution of a T12 tool and counting the pixels between the same feature one revolution apart. The measurements showed that the pixels were square.

4.2 Data set

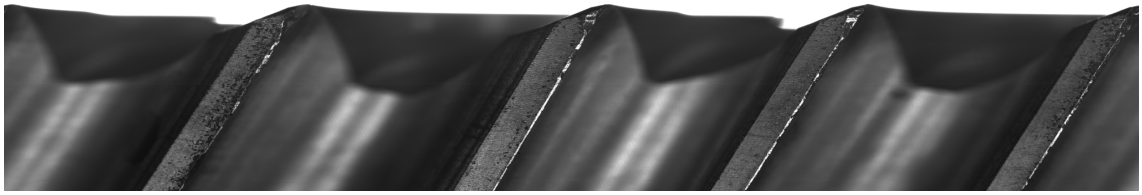
In total 200 images were collected from 57 tools. It took three images to cover the entire length of the T8 and T12 tools, the T10 tool required five images. Tab. 4.3 summarizes all images that were collected and the camera settings used. Lighting and aperture were the same for all pictures. An aperture of f/8 was used since it offered a good compromise between depth of field and light-transmitting ability. Using eq. 2.1 and assuming a circle of confusion of 4 pixels (28 µm), gives a depth of field of 0.35 mm. An ADC bit depth of 8 bits was used for all images. This is since the full 12-bit resolution was not possible with the large image size required to capture the end mills.

Changes were made to exposure time, image size, line rate, and rotation speed depending on the tool type. Image size and rotation speed defined the line rate and end mill and were calculated with eq. (4.1) and (4.2). The line rate was chosen depending on the exposure time. To keep the focus distance and magnification the same for all tool diameters the camera was moved depending on the tool.

Table 4.3: Summary of all images collected and the settings used. Quantity is the number of tools looked at, images is the number of images acquired, $\boldsymbol{\Omega}$ is their rotation speed, ET is the exposure time, and LR is the line rate.

Quantity	Tool	Images	Image size	Ω [deg/s]	ET [µs]	LR [Hz]
4	T8	12	2048x6960	25.9	1400	500
15	T10	74	2048x8700	41.4	400	1000
38	T12	114	2048x10440	34.5	400	1000

Fig. 4.4 shows examples of the images in the data set.



(a) Example image of the T12 tool.

(b) Example image of the T10 tool.

Figure 4.4: Examples of the images in the data set.

All images were annotated using VGG Image Annotator [57]. The areas marked in the annotation process were flank wear, fractures, and flank face. The flank face was added since it is the reference used for quantifying flank wear and fractures. Fig. 4.5 shows an example of an annotated flank face from one of the T12 tools.

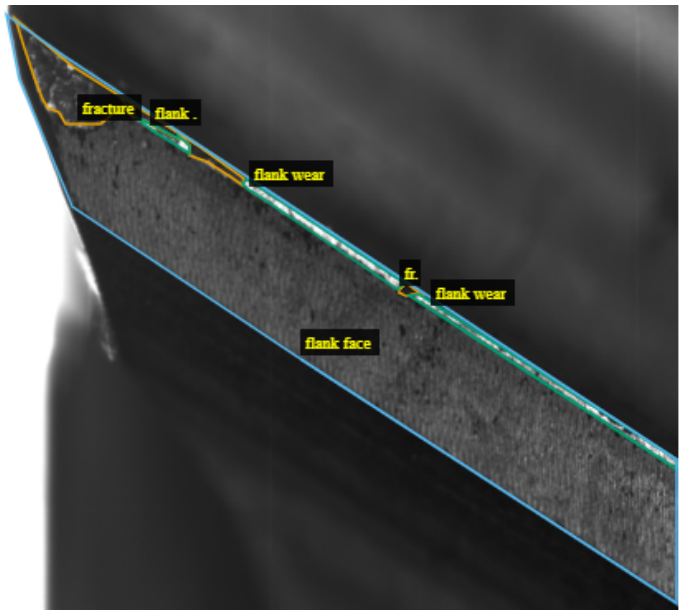


Figure 4.5: Annotated flank face of a T12 tool. The blue outline is the flank face, the green outline marks flank wear, and the orange outline marks fractures.

The images were split into three data sets for training, validation, and testing. The training set contained 127 images, this set was used for training all models. The validation set contained 35 images and was used when tuning hyperparameters and testing different settings. The test set contains the remaining 38 images, this was used to evaluate the final performance of the model. Images of T10 and T12 tools were split evenly across the three sets. All images of the T8 tools were added to the testing data set. This was done to test how the knowledge gathered from the T10 and T12 tools would transfer to the unseen T8. Tab. 4.4 summarizes the content of the three data sets.

Table 4.4: Summary of the data sets.

	Training	**Validation**	**Testing**
Number of images			
Total	127	35	38
T8 tool	0	0	12
T10 tool	49	14	11
T12 tool	78	21	15
Number of annotated areas			
Flank faces	704	200	233
Flank wear	1073	232	283
Fractures	824	197	234

4.3 Image Analysis

This section presents the steps taken to analyze the images and extract the final measurements.

4.3.1 Wear Segmentation

The first part of the image analysis was to extract a binary mask from the image that shows what pixels include wear. For this, the Mask R-CNN framework [44] was chosen. Mask R-CNN is an instance segmentation framework, making it appropriate for the task of segmenting instances of wear in the images. It was chosen since it is a popular framework with lots of freely available resources. Implementation was done using the Mask R-CNN class provided by the torchvision library [58].

The Mask R-CNN implementation included in the torchvision library can take an arbitrary-sized image as input. It accomplishes this by resizing the input to 800x800 pixels. The output from the model is a bounding box, a classification, and a mask for each instance it finds in the input image. The mask consists of a score for each pixel. The score is numbers between zero and one and show how confident the model is that a pixel should be included in the binary mask for that specific instance. A score from zero to one is also generated for each bounding box, the score shows how confident the network is that the bounding box contains an object of interest.

A ResNet-50 feature pyramid network (FPN) was used for the backbone. This is an FCN and was used since torchvision provides pre-trained weights for the ResNet-50 FPN backbone. The default ROI pooling provided by torchvision outputs a 7x7 feature map for classification and bounding box generation and a 14x14 feature map for generating the mask. In this case, the interesting features were often large relative to the image size, as seen in Fig. 4.7. Thus resizing all features to a 14x14 feature map might cause a bottleneck in the segmentation process.

Before and after the Mask R-CNN, pre-processing and post-processing were used to prepare the data. All the steps in the segmentation process are shown in Fig. 4.6.

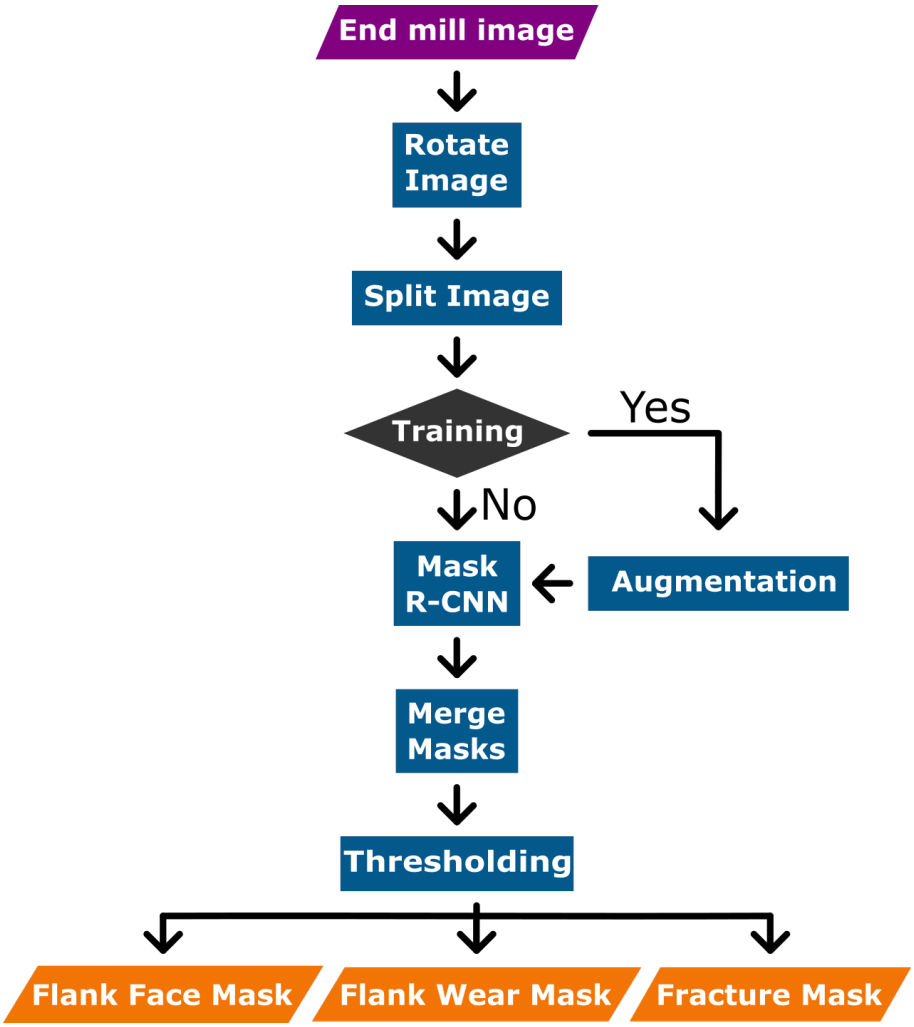
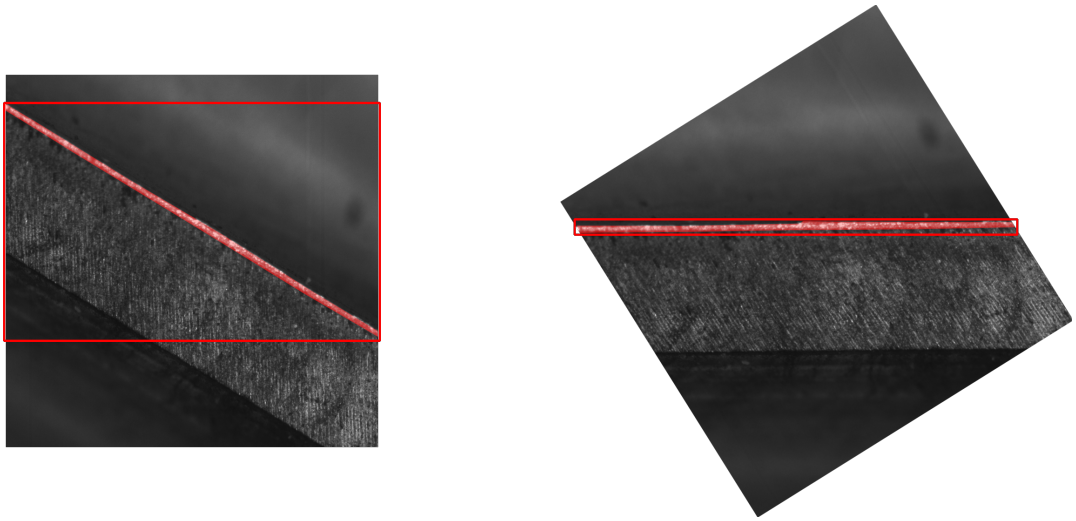


Figure 4.6: Flow chart that shows the components of the wear segmentation algorithm.

4.3.1.1 Pre-processing

As described in Sec. 2.4.3 the Mask R-CNN framework works by first finding the bounding boxes of an interesting region. The bounding boxes are aligned with the image axis. In the images of the end mills, the areas of interest are often thin and run diagonally over the image, as shown in Fig. 4.7a. This results in a bounding box that encompasses much more than the wear area. The ROI pooling step resizes all proposed regions to a fixed size, thus a larger region will lose more detail compared to a smaller region. For this reason, experiments were run where the input image was rotated by the end mill's helix angle, as in Fig. 4.7b. The rotation was applied in pre-processing and not by physically rotating the camera since rotating the camera introduces problems with the depth of field and distortion due to the cylindrical shape of the end mill.



(a) Mask and bounding box of flank wear without rotating the image.

(b) Mask and bounding box of flank wear after rotating wear without rotating the image.

Figure 4.7: The two images show how the bounding box is affected by rotating the image of the end mill.

The size of the end mill images was much larger than the 800x800 px input size of the Mask R-CNN model. Resizing the images would result in a loss of information. Thus, the same tiles strategy used by U-net [59] was adopted. The images are split into 800x800 px tiles that can be sent to the network without resizing. To not lose information at the edge of each tile, the tiles are overlapping as illustrated in Fig. 4.8. An overlap of 64 px was used resulting in a usable area for each tile of 672x672 px.

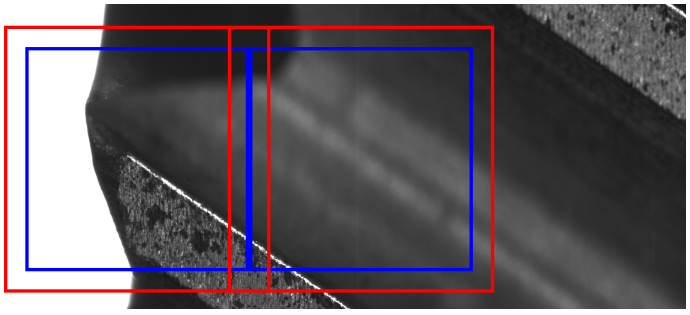


Figure 4.8: The figure shows the overlapping tiles. The red area shows the 800x800 px tiles and the blue area shows what is left of the tile after the overlap is removed and the image is reassembled.

The Mask R-CNN model expects a color image with RGB channels as input. Thus, the grayscale image was converted into an RGB image before passing them to the model.

4.3.1.2 Post-processing

The post-processing step takes the outputs from the Mask R-CNN and returns three binary masks, one showing the flank faces, one showing the flank wear, and one mask for the fractures. This mask had the same dimensions as the original image and was constructed by merging the mask from each image tile into one mask for each wear type.

The Mask R-CNN network does not directly output binary masks. Instead, it outputs predictions with a score for each bounding box and a score for every individual pixel in the masks. To convert this output into binary masks thresholds were applied, first to the bounding box score, and then to every individual pixel score in the mask. If the bounding box score was below the threshold value the corresponding mask was disregarded. Individual threshold values were used for the separate wear types, thus six threshold values were used in total.

The values for the thresholds were selected using grid-search [60] over the possible threshold values. The threshold values with the highest intersection over union were used for the algorithm.

4.3.1.3 Training

Transfer learning was used for training the Mask R-CNN network, due to the limited data set. Transfer learning is a technique where training results are transferred from one model to another model [61]. For the Mask R-CNN network, this was done by starting the training process using parameters trained on the Microsoft Common Objects in Context (COCO) data set [62]. When using transfer learning it is possible to only train some layers of the network. This is standard for the torchvision Mask R-CNN with a ResNet-50 backbone, where only three of the five backbone layers are trained. Due to the large differences between the COCO data set and the end mill data set all layers were trained in this case.

Data augmentation was used to extend the data set for training. The augmentation techniques were developed to emulate possible variations between different end mills. Since the measurement setup was controlled, the possible differences in the images are relatively small. Three effects were simulated with augmentation, these were helix angles, image brightness, and image contrast. The effects were applied using the PyTorch transform functions [63]. To simulate different helix angles a $\pm 3\,\text{deg}$ rotation was applied to the images when training. This was due to the focus on minimizing the bounding boxes, hence a big rotation would undo the rotation from the pre-processing. Fig. 4.9 shows examples of how the training images for the Mask R-CNN looked after the augmentation.

When training the Mask R-CNN network all image tiles that did not include a fracture, flank wear, or a flank face were removed from the data set. This significantly speeds up the training process by removing around 30 % of the image tiles for the image with no rotation in the pre-processing and 75 % for images when rotation

was applied. The big reduction in images for the rotated data set is due to padding that was added to the images when rotating them, thus many of the image tiles only contained padding in the rotated data set.

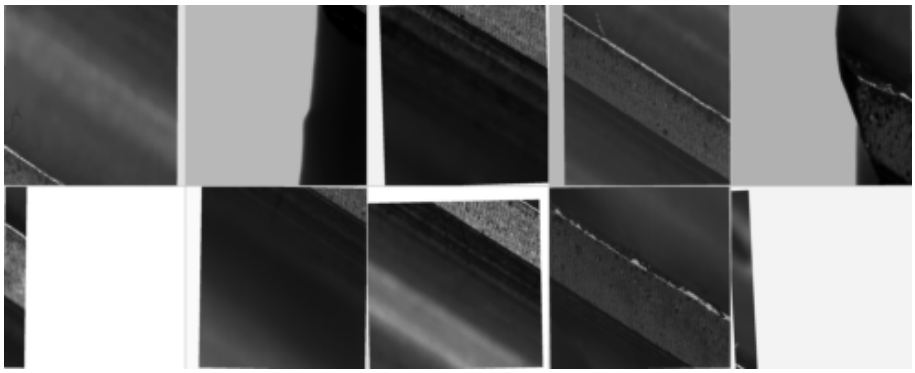


Figure 4.9: Example of image tiles in the training data set after augmentation.

4.3.2 Wear Quantification

The steps that are included in the method developed for quantifying the wear are shown in Fig. 4.10. As input the method takes the binary mask from the post-processing. The output is the average and maximum flank wear depth and the maximum fracture depth.

The first step of quantifying the wear was to extract the wear and flank face contours from the binary mask. This was done in order to remove unnecessary data, shrinking it from three binary images to a few polygons.

The wear depth should be measured from the unworn edge. Since the unworn edge no longer exists in the images it has to be approximated. This was achieved by approximating the flank face with the flank face mask from the segmentation step. The contour of the flank face mask was then used as the reference edge for all measurements.

After a reference edge was constructed the actual measurements could be extracted. This was done in a three-step process. As shown in Fig. 4.11a the contours did not always align with the reference edge. Since all measurements should be done from the reference edge, the first step was to extend the wear contour to the edge, as shown in Fig. 4.11b.

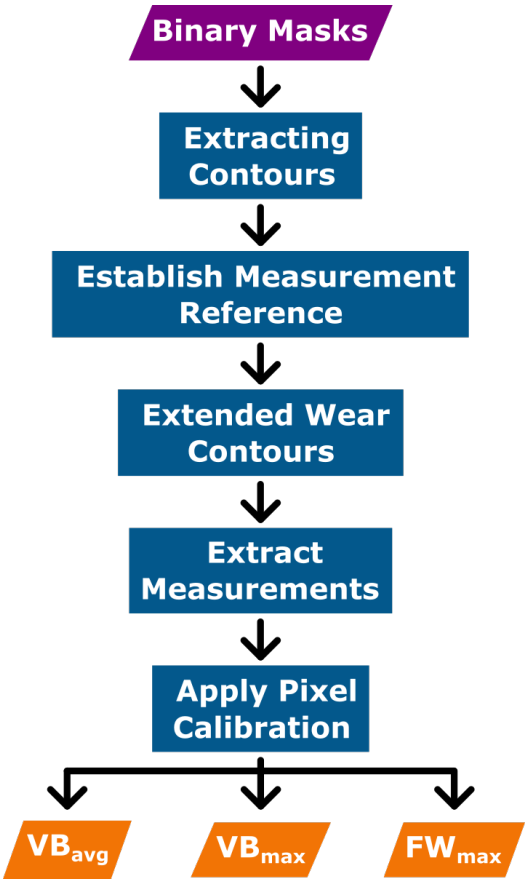
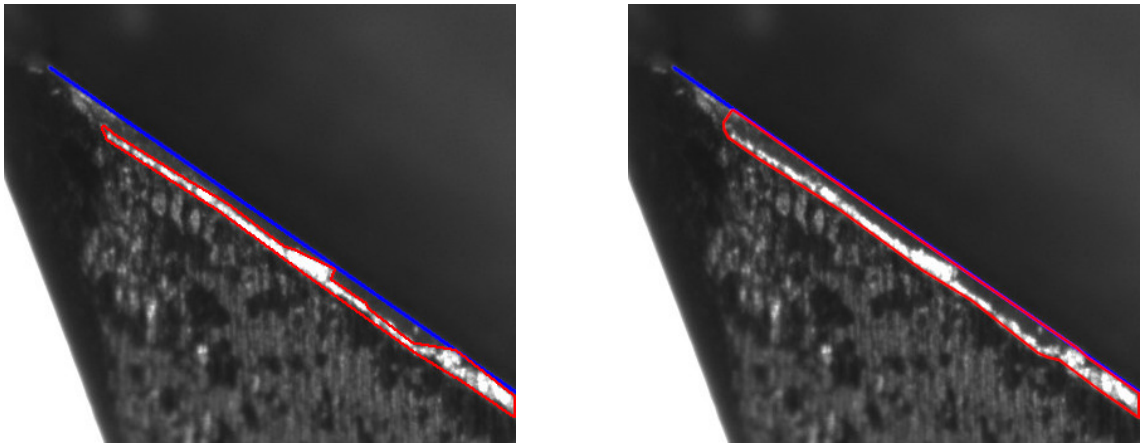


Figure 4.10: Flow chart that shows the components of the wear quantification algorithm.



(a) The red flank wear contour does not extend to the blue line that marks the cutting edge.

(b) The red flank wear contour from the left figure has been extended to include everything up to the cutting edge.

Figure 4.11: The wear contour in the left figure is extended to the cutting edge as in the right image.

The next step was to extract the measurements from the extended wear contour. The depth should be measured perpendicular to the edge, for this reason, the extended wear contour was rotated by the helix angle. When the contour is rotated, the maximum wear depth is the maximum difference in y values for the extended wear contour and the wear length is the difference in x values. The helix angle was measured from the flank face. The wear length was calculated as in eq. (4.5). The max depth was first calculated in small regions and the global max was extracted by taking the maximum from the smaller regions, as defined by eq. (4.4).
Finally, depending on if the contour is for flank wear or a fracture the corresponding measurements were extracted using eq. (4.6) and (4.7).

$$d_{max} = \max_{x \in ewc_x} (\max(ewc_y[x : x + a]) - \min(ewc_y[x : x + a])), \tag{4.4}$$

$$\ell = \max(ewc_x) - \min(ewc_x), \tag{4.5}$$

ewc_x and ewc_y is the x and y coordinates of the polygon defining the extended wear contour. d_{max} denotes the maximum wear depth and ℓ denotes the length of the wear area. $ewc_y[x : x + a]$ denotes all y points that have a corresponding x value in the range $[x, x + a]$, $a = 35$ µm was used.

$$VB_{max}, FW_{max} = kd_{max}, \tag{4.6}$$

$$VB_{avg} = k\frac{a}{\ell}, \tag{4.7}$$

w is the width of the minimum bounding box, a is the area of the wear contour and ℓ is the length of the minimum bounding box. k is the pixel resolution determined to be 3.6 µm/px in Sec. 4.1.3

4.3.3 Experiments

Experiments were done for both segmentation and quantification. All tests were run on the testing data set. The results were split into two categories. First was the result of the T10 and T12 tools, these tools were in the training and validation data set and had thus been used to train the networks. The other category where for the unseen T8 tools. The T8 tools were used to evaluate how the proposed methods generalized to unseen tool types. This is to get an idea of how the proposed methods could be adapted to new tools in the future.

When it comes to the segmentation method, three potential improvement areas were identified in Sec. 4.3.1, 4.3.1.1 and 4.3.1.3. For this reason, different Mask R-CNN models were trained to implement these different options. From here on the networks will be referred to as WSN (Wear Segmentation Network), with an added description. Tab. 4.5 shows the differences between the four versions.

Table 4.5: Differences between the four WSNs

Parameter	WSN Base	WSN Rot	WSN Aug	WSN Mask20
Initial Parameters	COCO	WSN Base epoch 15	WSN Base epoch 15	WSN Base epoch 15
Epochs	35	20	20	20
Pre-processing rotation	0°	33°	33°	33°
Augmentation	No	No	Yes	No
Pooling Output	7x7, bbox 14x14, mask	7x7, bbox 14x14, mask	7x7, bbox 14x14, mask	7x7, bbox 20x20, mask

As stated in Sec. 3.4, IoU, precision, and recall were used to evaluate the performance of the WSNs. Calculating precision and recall for segmentation masks requires a way of determining TP, FP, and FN. As stated in Sec. 2.3.2, IoU can be used for this, however in this case IoU is not a suitably metric. The reason is that it is not that important how the segmentation is split into different instances. If the target region is one instance and the WSN finds the same region but splits it into five instances it is still a good segmentation, the important part is finding the correct area. However, IoU will result in low scores in the case of five separate instances. To solve this a modified version of IoU was introduced, the metric is defined by eq. 4.8 and will be referred to as intersection over prediction (IoP). In contrast to the IoU, IoP returns the maximum score of 1 as long as the prediction is contained inside the target mask.

TP, FP, and FN wast then defined as using IoP. A prediction with an IoP lower than 0.5 was classified as FP. A target region was defined as detected (TP) if more than half of its area was covered by predictions not classified as FP. Otherwise, it was classified as an FN.

$$IoP = \frac{|P \cap T|}{|P|}, \tag{4.8}$$

where P is the predicted area and T is the target area.

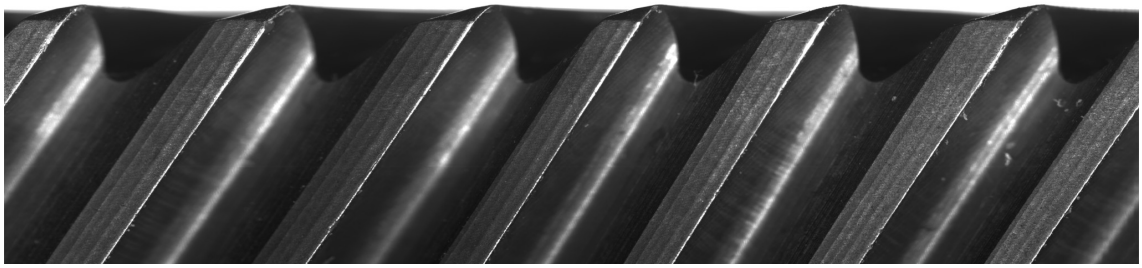
The quantification algorithm was applied to all the outputs from the WSNs in Tab. 4.5. To calculate the measured error a reference value is needed. This was calculated by applying the same quantification algorithm to the ground truth masks for each image tile. Thus, this assumes that the quantification algorithm can extract the correct values.

5

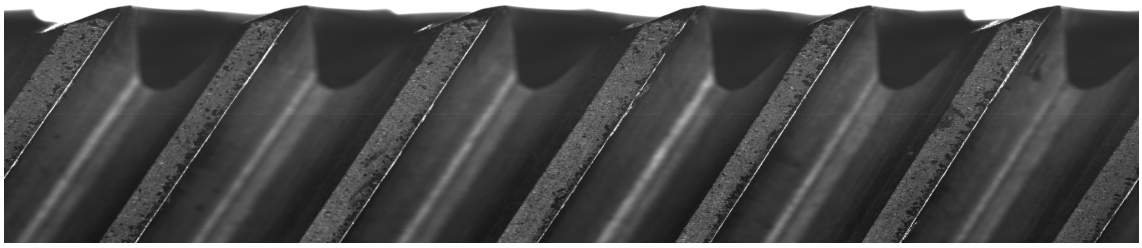
Result and Discussion

5.1 Image Acquisition

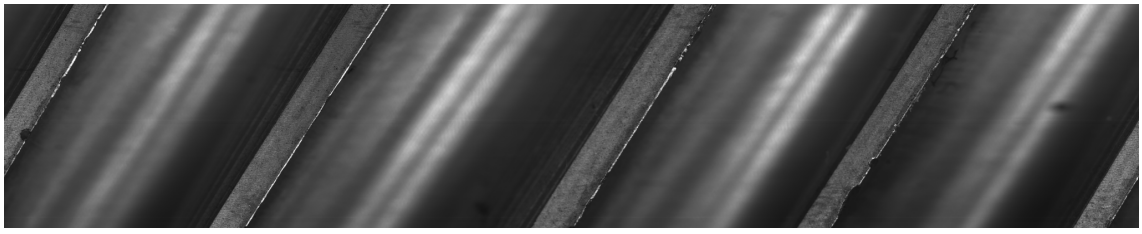
Fig. 5.1 shows the resulting images from the image acquisition setup. In all the images the flank wear shows up as a bright line on the edge, showing that the dark field illuminating fulfilled its purpose. This also makes it easier to detect fractures as they show up as darker parts on the edge. This can be seen in Fig. 5.1c where the edge is dark in many places due to small and larger fractures. Further, Fig. 5.1b and 5.1a shows how the backlight created a white background, resulting in a clear contour of the tool tips.



(a) Image of a T8 tool.



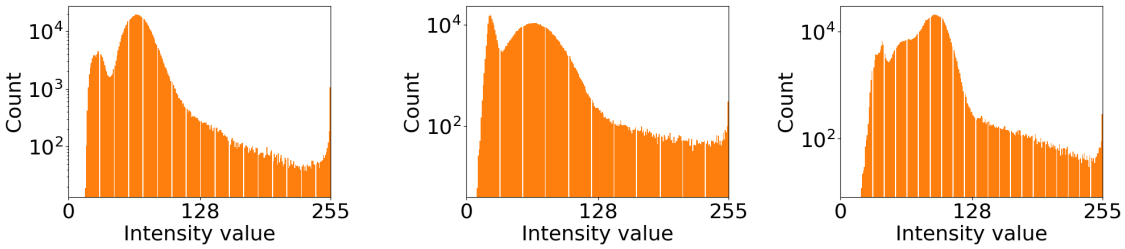
(b) Image of a T10 tool.



(c) Example image of a T12 tool.

Figure 5.1: Images captured of end mills.

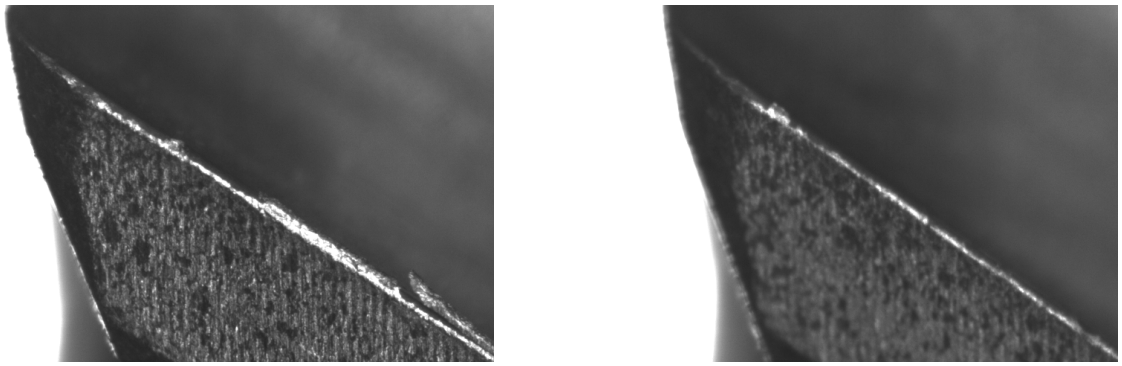
Fig. 5.2 shows the intensity distribution for the flank face of the different end mill variants. All of them show a spike at the maximum intensity of 255, indicating a slight overexposure. This is done to promote the shine from areas with flank wear.



(a) T8 tool. **(b)** T10 tool. **(c)** T12 tool.

Figure 5.2: Intensity histograms for the flank face of the three end mill variants.

One problem with the image acquisition setup was that the end mills moved in and out of focus when rotating them. Fig. 5.3 shows an example of an image where the focus changed when rotating the end mill. The reason is either that the tool was not properly centered on the rotating table or the end mill was leaning relative to the rotational axis. A run-out of 0.3 mm was measured for one of the T10 tools. With the narrow depth of field used, a 0.3 mm run-out is more than enough to move the end mill out of focus. This was not a problem for all end mills, thus likely being a consequence of not mounting the collet firmly to the rotating table. Noticeably the T12 tools did not seem to suffer from this problem, this is presumably since a different tool holder was used for the T12 tools.



(a) T10 tool tip in sharp focus. **(b)** T10 tool tip in poor focus.

Figure 5.3: Two image tiles of different flank faces form the same image of a T10 tool.

Time is an important consideration if this is going to be viable in an industrial application. With the camera and rotation settings used to collect the data set it took 14 s to capture an image of the T8 tool, 9 s for the T10 tool, and 11.5 s for the T12 tool. One image did not cover an entire end mill, thus the total time to capture all images for one end mill was 42 s for the T8 tool, 45 s for the T10 tool, and 34.5 s for the T12 tool. This time does not include the time it took to place

the end mill on the rotating table or the time for moving the camera into position. The limiting factor is the required exposure time which is limited by the amount of light reaching the camera. However, even without changing the lighting from the current setup, the time could likely be decreased by tuning the gain, exposure time, and rotation speed.

5.2 Mask R-CNN Wear Detection

Tab. 5.1 shows the resulting threshold values from the grid search. These thresholds were used in all further experiments.

Table 5.1: The result from parameter optimization on the validation data set.

Region	WSN Base	WSN Rot	WSN Aug	WSN Mask20
Bounding Box Threshold				
Flank face	0.47	0.82	0.95	0.69
Flank wear	0.60	0.95	0.46	0.82
Fracture	0.90	0.78	0.65	0.82
Mask Threshold				
Flank face	0.47	0.52	0.52	0.52
Flank wear	0.35	0.56	0.17	0.56
Fracture	0.43	0.39	0.43	0.39
Achieved Mean IoU				
Flank face	0.98	0.98	0.96	0.98
Flank wear	0.58	0.70	0.43	0.70
Fracture	0.50	0.50	0.37	0.53

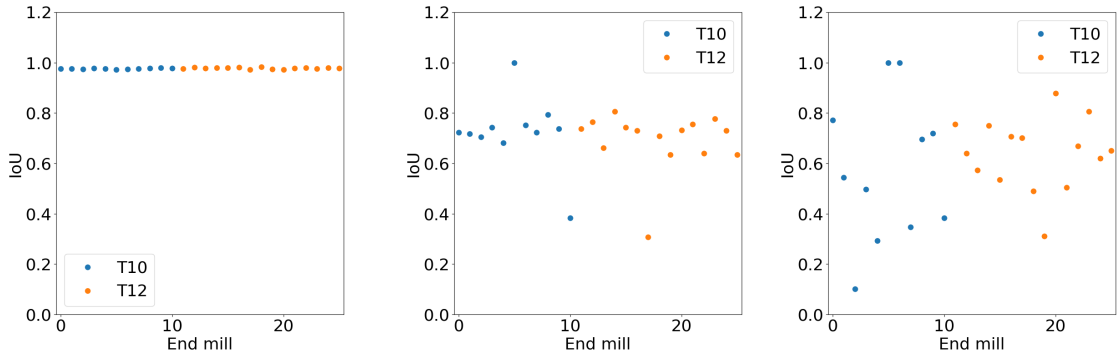
Tab. 5.2 shows the IoU for the different WSNs achieved. It is clear that all networks worked well for segmenting the flank face. Differences in the networks come up when comparing how they perform on segmenting the wear. Of the three changes made the most impact was achieved when rotating the images in the pre-processing step, this resulted in a significant increase to the mean IoU for both flank wear and fractures. Rotating was also the only change that significantly improved the performance of the networks. WSN Mask20 also improved on the base network, but noticeably it performed similarly to WSN Rot. Thus, increasing the output size from the ROI pooling did not result in a significant difference in performance. Comparing WSN Rot and WSN Mask20 shows that WSN Rot has a slightly better mean IoU for fractures, whereas WSN Mask20 has a tighter distribution of the flank wear IoU. Fig. 5.4 and Fig. 5.5 show the distribution of IoU for WSN Rot and WSN Mask20 on all T10 and T12 images in the testing data set. IoU distribution for all networks can be seen in appendix A. Both WSN Rot and WSN Mask 20 show similar results

as [12], for wear segmentation, although it is somewhat unclear what wear type [12] looked at, it is presumed to be flank wear.

The augmentation that was added when training WSN Aug, resulted in a significant decrease in performance. With performance being similar to or worse than the base network in all IoU categories. Flank face IoU show by far the most significant decrease. The reason for this decrease could be a combination of the lighting and the slightly overexposed images. This combination makes the flank wear show up as a bright line in the images. However, this combination also leads to a lack of texture and detail in areas with flank wear since most pixels will be saturated. Thus, it is feasible that the neural network relies heavily on brightness to determine where flank wear exists in an image. Hence, modifying the brightness and contrast in the augmentation step would make it harder to detect flank wear. As seen in Fig. 5.6 WSN Aug finds the location of the flank wear, however, it has a hard time classifying individual pixels. As an example, the annotated flank wear closest to the tip in Fig. 5.6 includes an area of the flank face that is too dark to be flank wear, but since the augmentation, sometimes results in much darker flank wear, the model can not detect that it is too dark.

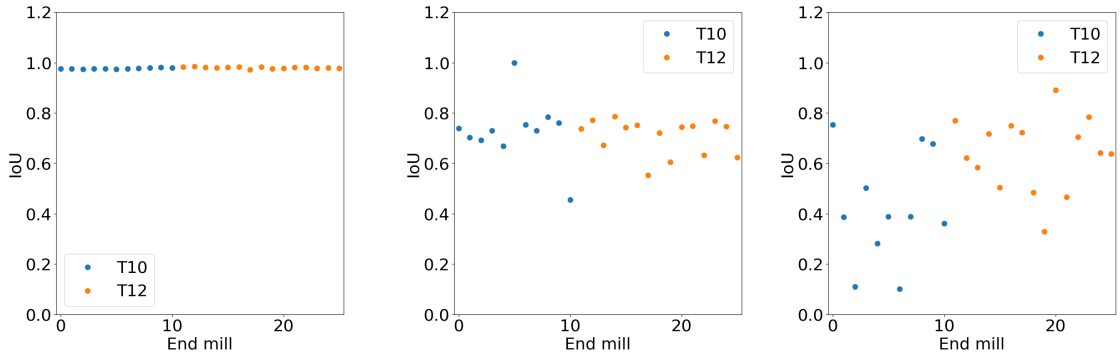
Table 5.2: Image segmentation results for the T10 and T12 tools in the testing data set.

Region	WSN Base	WSN Rot	WSN Aug	WSN Mask20
Mean IoU				
Flank face	0.98	0.98	0.97	0.98
Flank wear	0.57	0.70	0.42	0.72
Fracture	0.55	0.61	0.53	0.55
Maximum IoU				
Flank face	0.98	0.98	0.98	0.98
Flank wear	1.00	1.00	1.00	1.00
Fracture	1.00	1.00	1.00	0.89
Minimum IoU				
Flank face	0.97	0.97	0.92	0.97
Flank wear	0.36	0.31	0.29	0.45
Fracture	0.05	0.10	0.09	0.10
IoU Standard deviation				
Flank face	0.00	0.00	0.01	0.00
Flank wear	0.12	0.13	0.17	0.10
Fracture	0.24	0.21	0.25	0.21



(a) Flank face. **(b)** Flank Wear. **(c)** Fracture.

Figure 5.4: IoU distribution for WSN Rot on images of the T10 and T12 tools in the testing data set



(a) Flank face. **(b)** Flank Wear. **(c)** Fracture.

Figure 5.5: IoU distribution for WSN Mask20 on images of the T10 and T12 tools in the testing data set

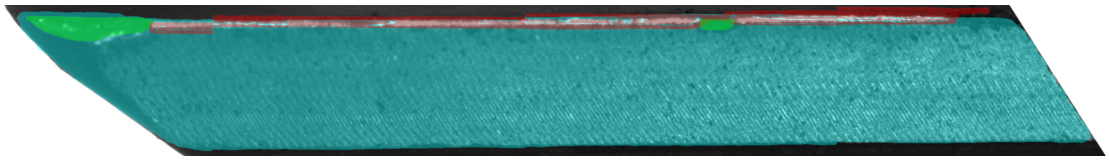


Figure 5.6: Annotation generated by WSN Aug for one flank face of a T12 tool. Blue areas are annotated as flank face, red areas are flank wear, and green are fractures.

Calculating the precision and recall, show the same trends as the IoU. With WSN Rot and WSN Mask20 having the best preference, as seen in Tab. 5.3. WSN Rot performs better with respect to precision, and WSN Mask20 performs better with respect to recall.

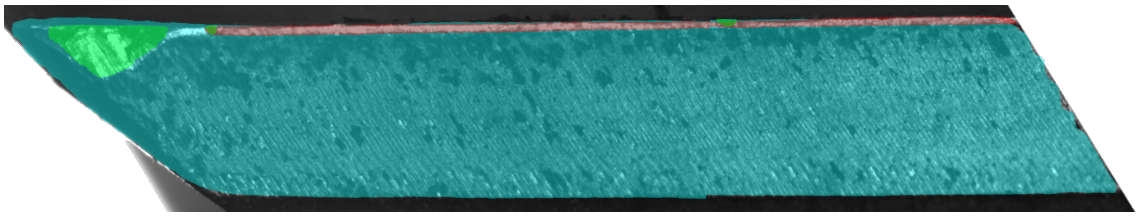
Exclude image tiles that did not include a part of the flank face from training does not seem to have impacted the number of false positives. All areas annotated by

the WSN were in the vicinity of a flank face, thus there were no problems with false positives that occurred in a part of the images not included when training.

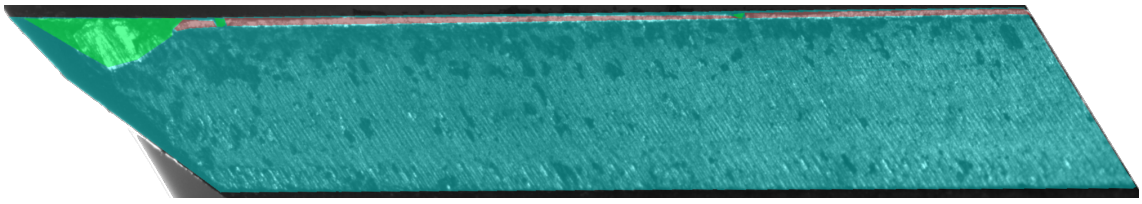
Table 5.3: Precision and recall results for the T10 and T12 tools in the testing data set.

Region	WSN Base	WSN Rot	WSN Aug	WSN Mask20
Precision				
Flank face	1.00	0.95	0.82	0.97
Flank wear	0.40	0.95	0.16	0.76
Fracture	0.68	0.79	0.61	0.77
Recall				
Flank face	1.00	1.00	1.00	1.00
Flank wear	0.78	0.87	0.17	0.92
Fracture	0.60	0.61	0.44	0.66

Fig. 5.7 and 5.8 show a comparison of the mask generated by WSN Rot and manual annotation for the T12 and T10 tools. For both tools, the network performs well and detects the major features. Defects that can be seen are a missed area of flank wear at the tip and overlap between the flank wear and the right most fracture in Fig. 5.7a. There are also some differences in the fracture annotating between Fig. 5.8a, and 5.8b.

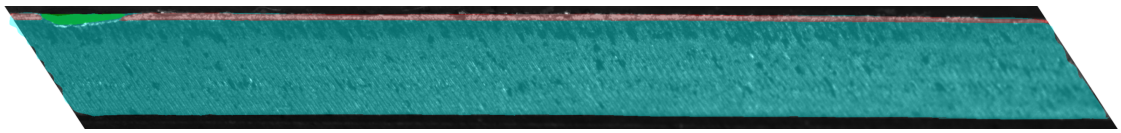


(a) Mask generated by WSN Rot.

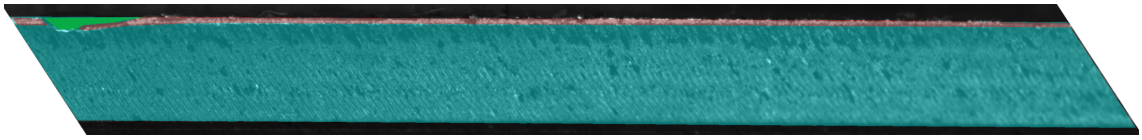


(b) Manual annotation.

Figure 5.7: Mask and manual annotation for one flank face of a T12 tool. Blue areas are annotated as flank face, red areas are flank wear, and green are fractures.



(a) Mask generated by WSN Rot.



(b) Manual annotation.

Figure 5.8: Mask and manual annotation for one flank face of a T10 tool. Blue areas are annotated as flank face, red areas are flank wear, and green are fractures.

5.3 Wear Quantification

Quantification results are shown in Tab. 5.4. The result follows the same pattern as the segmentation results in that WSN Base and WSN Aug performed the worst and WSN Rot and WSN Mask20 performed the best. A notable difference between WSN Rot and WSN Mask20 is in the quantification of maximum fracture depth, where WSN Mask20 had a better average performance. This is surprising since in the segmentation WSN Rot had a higher IoU compared to WSN Mask20 for segmenting fractures. Looking at the maximum percentage error it is much lower for the WSN Mask20 compare to WSN Rot, however, the maximum error is higher for WSN Mask20. Thus, WSN Mask20 might not be better in general but performed better in some extreme cases with small fractures. As seen in Fig. 5.9 WSN Rot had two outliers with a 100 % error. In both cases, this was for images with small fractures, where the fractures were not detected by the WSN.

The best mean absolute flank face error achieved was 12 % for WSN Mask20, this is high compared to [13, 14, 51], who all present a percentage error in the range of 3 % to 7 %. Part of the higher error in this work can be explained by how the reference edge is established. The impact of the reference edge was tested by swapping the flank face annotating from the WSN with the manual annotation in the quantification algorithm. Since the flank face annotation is used to establish the reference edge this is the same as manually defining the reference edge for every flank face in every image. For WSN Rot and WSN Mask20, this resulted in both networks showing a 7 % mean absolute error for the VB_{avg} measurement, a significant improvement. An improvement was also seen for FW_{max} with a mean absolute error of 19 % and 15 % for WSN Rot and WSN Mask20, respectively. Surprisingly it had no major effect on VB_{max}, which increased by two percentage points for WSN Rot and decreased by one percentage point for WSN Mask20.

Table 5.4: Quantification results for the T10 and T12 tools in the testing data set.

Region	WSN Base		WSN Rot		WSN Aug		WSN Mask20	
	µm	%	µm	%	µm	%	µm	%
Mean Absolute Error								
VB_{avg}	15	35	6	13	8	15	6	12
VB_{max}	25	32	15	16	32	39	17	18
FW_{max}	30	20	29	24	50	39	24	17
Max Absolute Error								
VB_{avg}	29	75	20	34	42	61	22	36
VB_{max}	93	148	39	40	68	94	43	58
FW_{max}	138	83	82	100	303	200	101	53
	Count		Count		Count		Count	
Absolute Error < 5%								
VB_{avg}	0 of 26		7 of 26		7 of 26		7 of 26	
VB_{max}	3 of 26		3 of 26		3 of 26		3 of 26	
FW_{max}	4 of 26		2 of 26		1 of 26		5 of 26	
Absolute Error > 15%								
VB_{avg}	22 of 26		12 of 26		6 of 26		9 of 26	
VB_{max}	16 of 26		12 of 26		20 of 26		14 of 26	
FW_{max}	12 of 26		14 of 26		20 of 26		14 of 26	

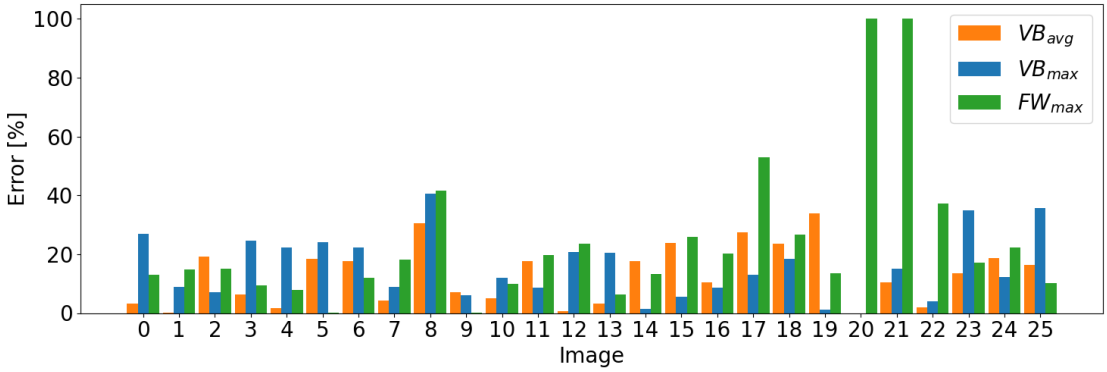


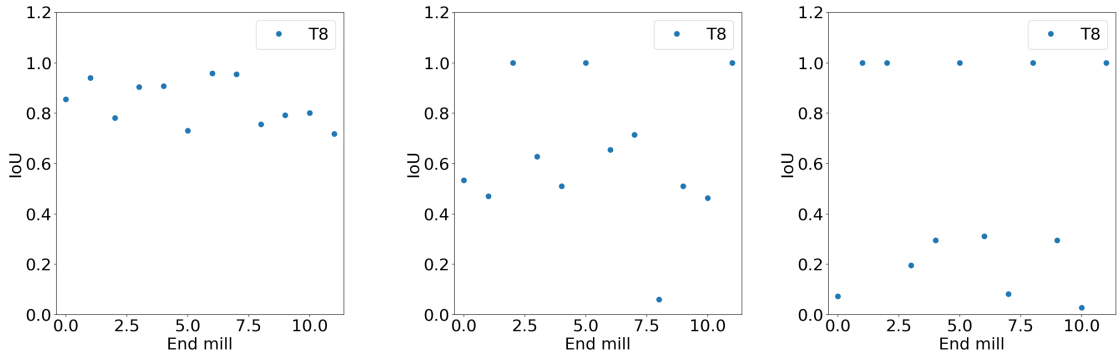
Figure 5.9: Distribution of the relative absolute error for quantification of T10 and T12 tools in the testing data set. The quantification is done with masks generated by WSN Rot. If the error is shown to be zeros it is because that wear type did not exists in the image (Thus it does not indicate a perfect measurement).

5.4 Adaptability to new Tools

Adaptability to new tools was tested by applying the algorithm to the T8 tools. Tab. 5.5 shows the IoU for the different networks and Tab. 5.6 shows the precision and recall. The results follow the same pattern as for the T10 and T12 tools, with rotation in the pre-processing improving the IoU. Similarly, augmentation had a detrimental effect on the segmentation. As seen in Fig. 5.10c the images are split into two groups, either the IoU is one or it is below 0.4. In the case where IoU is one, it is because the images do not contain any fractures. Thus, the mean IoU of 0.52 could be considered to be misleadingly high, since the IoU when there are fractures in the image is significantly lower than 0.52. The same partly holds for flank wear but it is not as extreme.

Table 5.5: Image segmentation results for the T8 tools in the testing data set.

Region	WSN Base	WSN Rot	WSN Aug	WSN Mask20
Mean IoU				
Flank face	0.81	0.85	0.83	0.84
Flank wear	0.52	0.63	0.43	0.64
Fracture	0.55	0.52	0.55	0.60
Maximum IoU				
Flank face	0.94	0.96	0.97	0.96
Flank wear	1.00	1.00	1.00	1.00
Fracture	1.00	1.00	1.00	1.00
Minimum IoU				
Flank face	0.64	0.72	0.62	0.70
Flank wear	0.02	0.06	0.02	0.05
Fracture	0.00	0.03	0.02	0.03
IoU Standard deviation				
Flank face	0.10	0.09	0.11	0.09
Flank wear	0.30	0.26	0.34	0.27
Fracture	0.40	0.40	0.45	0.41



(a) Flank face. **(b)** Flank Wear. **(c)** Fracture.

Figure 5.10: IoU distribution for WSN Rot on images of the T8 tools in the testing data set

Compared to the T10 and T12 tools, experiments on the T8 tool showed that annotation outside the flank face was a problem. An effect of this is a decrease in precision for flank face segmentation, as seen in Tab. 5.6. This might be improved by training the networks on all image tiles, thus not removing tiles without a part of the flank face.

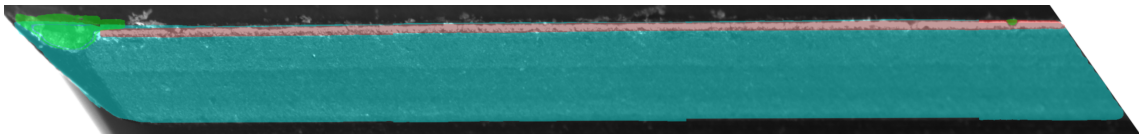
Table 5.6: Precision and recall for image segmentation on the T8 tool from the testing data set.

Region	WSN Base	WSN Rot	WSN Aug	WSN Mask20
Precision				
Flank face	0.78	0.74	0.65	0.75
Flank wear	0.13	0.62	0.03	0.46
Fracture	0.23	0.38	0.11	0.30
Recall				
Flank face	0.95	0.93	0.95	0.95
Flank wear	0.35	0.70	0.13	0.70
Fracture	0.15	0.19	0.07	0.14

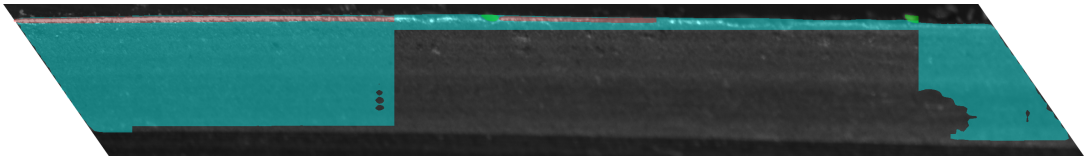
Fig. 5.11 shows how the performance of the annotation varied for the T8 tools. From a good annotation as seen in Fig. 5.11a, where all of the fractures and flank wear are detected, to a bad annotation as seen in Fig. 5.11b and 5.11c. In Fig. 5.11b large areas of both the flank face and flank wear are missing from the annotation. Another problem is that areas outside the flank face were often categorized as either flank face, flank wear, or a fracture. Examples of this can be seen in both 5.11b and 5.11c, where all the fractures are found outside the flank face.

It is worth noting that many of the problems with the annotation of the T8 tools will not affect the quantification result. For example, the annotation in Fig. 5.11b

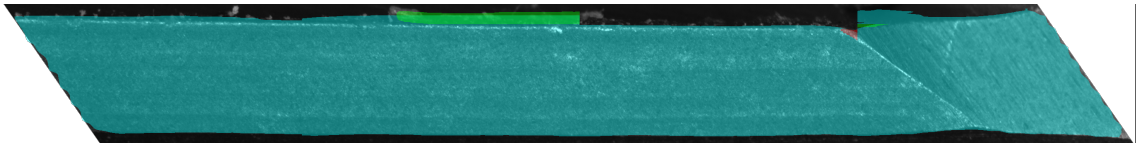
will likely give accurate results after quantification. This is since the annotate flank face does not extend far outside the real flank face and all the wear areas that are found inside the annotated flank face are correct. Thus both the maximum wear depth and average will be calculated correctly over the found wear areas. With even wear over the entire edge, the localized measurements will be a good approximation for the whole edge. However, this does not hold for Fig. 5.11c where the annotate flank face area is too big, thus the fractures found outside the flank face will still be measured. Fig. 5.11c corresponds to bars labels as *image 2* in Fig. 5.13, there it is possible to see a big FW_{max} error caused by a too large flank face annotation.



(a) Example of a good mask, all fractures and flank wear are correctly annotated. However, the area marked as a fracture at the tip is too large.



(b) Example of a bad mask. Missed a lot of the flank face and flank wear. Annotated area outside the flank face as a fracture.



(c) Example of a bad mask. Missed a lot of the flank face and flank wear. Annotated area outside the flank face as a fracture.

Figure 5.11: Masks generated by WSN Rot for three different flank faces from two T8 tools. Blue areas are annotated as flank face, red areas are flank wear, and green are fractures.

Tab. 5.7 presents the result for wear quantification on the T8 tools. It shows that WSN Rot has a clear advantage over the other networks, performing better or similar in all categories. Still, the performance is far from what was presented for the T10 and T12 tools. As discussed previously the quantification algorithm should be robust enough to deal with some imperfections, however, this was not enough. One big problem was that the flank face mask was often far bigger than the actual flank face, thus moving the reference for all the measurements. As shown in Fig. 5.12, the large maximum errors of 1860 % for WSN Aug were caused by a too large flank face annotation. The same holds for the maximum errors of 1020 %, and 1270 %. A test was done using the manually annotated flank face to establish a reference edge. Using WSN Rot this resulted in a mean absolute error of 16 % and 20 % for VB_{avg} and VB_{max}, respectively. However, the mean absolute error for FW_{max} only decreased by two percentage points.

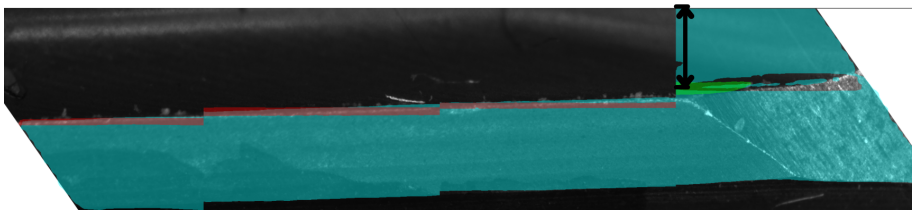


Figure 5.12: Masks generated by WSN Aug for a flank face of a T8 tool. Blue areas are annotated as flank face, red areas are flank wear, and green are fractures. The black arrow shows what is measured as the maximum fracture depth.

Table 5.7: Quantification results for the T8 tools in the testing data set.

Region	**WSN Base**		**WSN Rot**		**WSN Aug**		**WSN Mask20**	
	µm	%	µm	%	µm	%	µm	%
Mean Absolute Error								
VB_{avg}	47	83	25	48	49	89	32.4	49
VB_{max}	124	103	42	40	284	245	170	155
FW_{max}	100	171	42	48	289	309	62	81
Max Absolute Error								
VB_{avg}	93	105	63	71	122	252	78	111
VB_{max}	465	360	93	75	1290	1020	1270	1000
FW_{max}	426	867	111	177	1190	1860	132	212
	Count		Count		Count		Count	
Absolute Error $<$ 5%								
VB_{avg}	0 of 12		0 of 12		0 of 12		0 of 12	
VB_{max}	0 of 12		0 of 12		1 of 12		1 of 12	
FW_{max}	0 of 12		1 of 12		0 of 12		1 of 12	
Absolute Error $>$ 15%								
VB_{avg}	9 of 12		9 of 12		9 of 12		8 of 12	
VB_{max}	9 of 12		8 of 12		8 of 12		8 of 12	
FW_{max}	8 of 12		8 of 12		7 of 12		9 of 12	

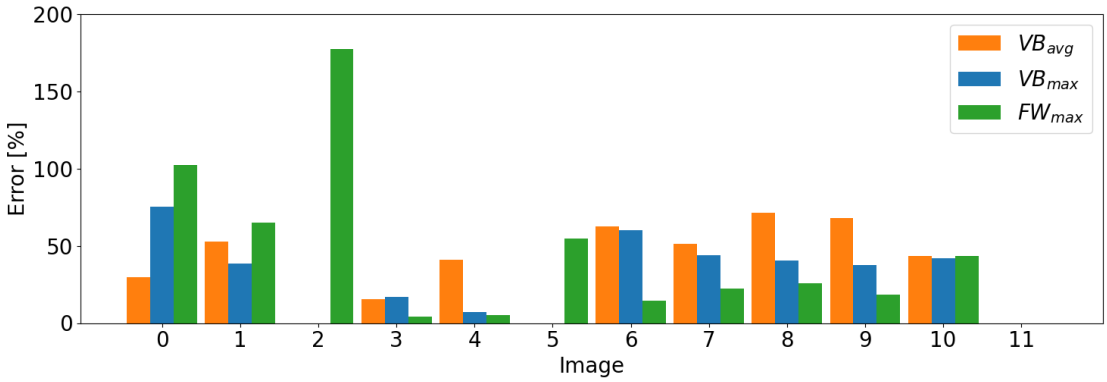


Figure 5.13: Distribution of the relative absolute error for quantification of T8 tools in the testing data set. The quantification is done with masks generated by WSN Rot. If the error is shown to be zeros it is because that wear type did not exist in the image (Thus it does not indicate a perfect measurement).

6
Conclusion

This thesis proposes a method for automatic offline tool wear measurement on end mills. The method is based on capturing and analyzing images of end mills. This is achieved with a camera rig that can collect images of end mills and input the images to a machine-learning-based computer vision algorithm. To train and test the algorithm 57 end mills of three different types were used. All end mills were at the end of their usable life and had been used to machine Haynes 282.

A camera rig was used to collect images around the tool circumferences. This was achieved with a line scan camera and rotating the end mills. End mills were mounted in a tool holder that was then placed on a rotating table. This made it possible to quickly change the end mills. A drawback of not firmly mounting the tool holder to the rotating table was that the end mill could be out of alignment with the rotational axis. This combined with a small depth of field resulted in some shifts in the sharpness of the images. Except for this, the setup performed well and was easily adapted to work for the different end mill types.

The core of the image analysis algorithm was a deep learning image segmentation network. For this, the Mask R-CNN framework was used. The Mask R-CNN was used to annotate important features in the image, these were the flank face, flank wear, and fractures. A few different variants of the image segmentation step were tested, the most significant change proved to be rotating the images before sending them to the Mask R-CNN. The rotation was done so that the wear area was horizontal in the image, thus minimizing the size of a bounding box around the wear. The best network achieved an IoU of 0.98 for detecting the flank face, 0.70 for the flank wear, and 0.61 for detecting fractures. The flank wear value of 0.70 is similar to the results presented by a recent study. The method was also tested on end mills, not in the training data set. This gave an IoU of 0.85 for detecting the flank face, 0.63 for the flank wear, and 0.52 for detecting fractures. Significantly lower, but still promising if the relatively large difference between the end mills is considered.

Segmentation was followed by a quantification algorithm where the maximum and average flank wear depth, as well as the maximum fracture depth, were extracted from the wear segmentation. Measurements were done using the flank face mask as a reference for the edge, the detected wear areas were extended to that edge and then measured. The results show a 13 % mean absolute error for the average flank wear depth, 16 % mean absolute error for the maximum flank wear depth, and a 24 % mean absolute error for the maximum fracture depth. This is a significantly

larger error than some other publications. However, due to the large variations in cutting tools, it is hard to draw any conclusions from this, notably, none of the publications used for comparisons look at the entire circumference of an end mill. Further, it is suggested that the error can be reduced significantly by establishing a better reference for the measurements. Something that should be possible since the geometry of an unworn flank face is.

In conclusion, the methods proposed are able to detect, classify and quantify wear on end mills. The results shown are promising, however, further research is needed to evaluate if and how this could improve tool utilization and machining processes in an industrial environment.

7
Further Work

Performance is an important aspect if a system like the one proposed by this thesis is to be viable in an industrial application. In terms of measurement accuracy, the results presented in this thesis are worse than the results presented in other publications. Some areas for potential improvements were identified. One problem with the proposed image acquisition setup was shifting focus when rotating the end mills. This could either be solved by improving the mounting of the end mills or by increasing the depth of field. For image segmenting, the reliance on deep learning makes it difficult to identify specific problems. A few different options were tested for training and configuring the Mask R-CNN, but this is an area where more research could be done. Another possible way of improving the segmentation could be a more advanced post-processing step that removes or modifies unreasonable annotations. One example is the fracture annotated in Fig. 5.12, it is impossible for a fracture to be that far down on a tool, thus it could be removed. Another example is to not allow flank wear and fracture annotations to overlap, it is impossible to have flank wear and a fracture in the same place.

As discussed in Sec. 5.3, the reference edge has a big impact on the result from the wear quantification step. The method of establishing a reference from only the segmentation leaves out a lot of knowledge about the flank face. The exact geometry of an unworn flank face is already known since all unworn tools of the same type have the same geometry. The only unknown parameter is the rotation of the end mill at the time the image was collected. Thus, by either measuring the angular position of the tool or extracting it from the image, a more accurate reference edge should be possible.

Establishing an accurate edge is also connected to the limitation of only looking at the part of an end mill that fits in one image. During this work, multiple images were taken of every end mill. The images were taken at different positions along the end mill and analyzed separately. However, to better describe the wear on a tool, it would be advantageous to look at an entire tool, instead of sections of the tool. For this, the relation between different images must be known. The camera position is already known, what is needed is the relative angular position of the end mill. If an accurate reference edge is constructed that edge could be used to determine the angular relation between images, this assumes that the tool is not rotationally symmetric. Another option is to measure the angular position of the rotating table when each picture is taken.

One of the goals set out from the beginning was that new tools should be easily added to the model. In the case of adaptability, the methods proposed in this thesis showed some potential when applied to a tool not included in the training data. However, this requires more work to verify the results on a more diverse set of tools and to improve the accuracy of unseen tools. One way of doing this could be with data augmentation. Some simple augmentation was tried in this work, but this resulted in worse accuracy, both for segmentation and quantification. A method for artificially adding wear to an unworn end mill would be a great way of simplifying the process of adding new tools since it would limit the time it takes to collect a data set for a new tool. This was one of the limiting factors during this work since annotating the images took a significant time.

With the limitations put on this thesis, a natural extension is to apply and adapt similar methods to a more diverse set of cutting tools. Both to verify the results on more end mill variations and to adapt the methods for other types of cutting tools and wear types. In adapting the proposed methods for other tool types, there are a lot of possibilities. Firstly, the proposed methods could most likely be directly applied to other tools with similar geometry and design, examples are ball end mills, corner radius end mills, drills, reamers, and roughing end mills. This would most likely require retraining of the neural networks for some of the tools. The same holds for other types of wear. By directly applying the proposed methods, the analyzed area would be limited to the cylindrical part of the tool. Thus, another option is to develop an image acquisition method that would allow capturing other areas of the tool. For standard end mills and roughing end mills, this could be done with an area camera aimed at the tool's end face. Other tools like drills and ball end mills would provide a bigger challenge due to their more complex geometry. A static area camera with an adequate depth of field could still work, but this would require the development of an algorithm that deals with distortion from projecting the end mill onto a flat plane. Another option is to take a similar approach to the method proposed in this thesis, where the camera is moved relative to the end mill.

Finally, integrating the proposed methods into an industrial environment is something that should be investigated. For this, the image acquisition and image analysis must be integrated into existing infrastructure. Further, workflows for how to deal with the acquired data and where to apply the data need to be developed. Here there are many possibilities. One option is to use the information to optimize tool life, possibly leading to a longer tool life while maintaining a good quality of the final parts. Other possibilities include correlating tool wear with part quality, in this way measuring tool wear could be a part of quality control. Similarly, problems with machines could possibly be detected by unusual tool wear, thus minimizing the time it takes to identify a problem.

Bibliography

[1] Aprilian Puji Pranoto. *Metal Cutting Theory And Practice 3rd by D. A. Stephenson*. CRC Press, 01 2016.

[2] Dominik Brenner, Fabian Kleinert, Joachim Imiela, and Engelbert Westkämper. Life cycle management of cutting tools: Comprehensive acquisition and aggregation of tool life data. *Procedia CIRP*, 61:311–316, 2017.

[3] Fredrik Schultheiss, Jinming Zhou, Elias Gröntoft, and Jan-Eric Ståhl. Sustainable machining through increasing the cutting tool utilization. *Journal of Cleaner Production*, 59:298–307, 2013.

[4] Keyvan Hosseinkhani and Eu-Gene Ng. A unique methodology for tool life prediction in machining. *Journal of Manufacturing and Materials Processing*, 4(1), 2020.

[5] E. Usui, T. Shirakashi, and T. Kitagawa. Analytical prediction of cutting tool wear. *Wear*, 100(1):129–151, 1984.

[6] Nitin Ambhore, Dinesh Kamble, Satish Chinchanikar, and Vishal Wayal. Tool condition monitoring system: A review. *Materials Today: Proceedings*, 2(4):3419–3428, 2015. 4th International Conference on Materials Processing and Characterzation.

[7] International Standard Organization. Test conditions for numerically controlled turning machines and turning centres – part 1: Geometric tests for machines with horizontal workholding spindle(s), 2020.

[8] Test conditions for numerically controlled turning machines and turning centres – part 2: Geometric tests for machines with a vertical workholding spindle, 2020.

[9] A Karthik, S Chandra, B Ramamoorthy, and S Das. 3d tool wear measurement and visualisation using stereo imaging. *International Journal of Machine Tools and Manufacture*, 37(11):1573–1581, 1997.

[10] Joakim Sandberg. Development of a laboratory test method for assessment of crater wear volume on inserts for steel turning, 2019.

[11] Xuefeng Wu, Yahui Liu, Xianliang Zhou, and Aolei Mou. Automatic identification of tool wear based on convolutional neural network in face milling process. *Sensors*, 19:3817, 09 2019.

[12] Thomas Bergs, Carsten Holst, Pranjul Gupta, and Thorsten Augspurger. Digital image processing with deep learning for automated cutting tool wear detection. *Procedia Manufacturing*, 48:947–958, 2020. 48th SME North American Manufacturing Research Conference, NAMRC 48.

[13] Avinash A. Thakre, Aniruddha V. Lad, and Kiran Mala. Measurements of tool wear parameters using machine vision system. *Modelling and Simulation in Engineering*, 2019:1–9, 2019.

[14] Ruitao Peng, Haolin Pang, Haojian Jiang, and Yunbo Hu. Study of tool wear monitoring using machine vision. *Automatic Control and Computer Sciences*, 54(3):259–270, 2020.

[15] Endmill selection guide. `https://www.redlinetools.com/resources/endmill-information`.

[16] Muhammad Wasif, Syed Amir Iqbal, Aqeel Ahmed, Muhammad Tufail, and Mahmoud Rababah. Optimization of simplified grinding wheel geometry for the accurate generation of end-mill cutters using the five-axis cnc grinding process. *The International Journal of Advanced Manufacturing Technology*, 105(10):4325–4344, 2019.

[17] Insert wear tool wear. `https://wetransfer.com/downloads/4a800d6064e9627cb7d315316a9654c520220504084613/b6b62b`.

[18] Wear on cutting edges. `https://www.sandvik.coromant.com/en-gb/knowledge/materials/pages/wear-on-cutting-edges.aspx`.

[19] Robin Jenkin. Chapter 9 - image sensors. In Elizabeth Allen and Sophie Triantaphillidou, editors, *The Manual of Photography (Tenth Edition)*, pages 155–173. Focal Press, Oxford, tenth edition edition, 2011.

[20] Jürgen Beyerer, Fernando Puente León, and Christian Frese. *Machine vision: Automated visual inspection: Theory, practice and applications*. Springer, 2015.

[21] Efthimia Bilissi, Sophie Triantaphillidou, and Elizabeth Allen. Chapter 12 - exposure and image control. In Elizabeth Allen and Sophie Triantaphillidou, editors, *The Manual of Photography (Tenth Edition)*, pages 227–243. Focal Press, Oxford, tenth edition edition, 2011.

[22] Sidney Ray. Chapter 6 - photographic and geometrical optics. In Elizabeth Allen and Sophie Triantaphillidou, editors, *The Manual of Photography (Tenth Edition)*, pages 103–117. Focal Press, Oxford, tenth edition edition, 2011.

[23] Elizabeth Allen and Efthimia Bilissi. Chapter 14 - digital cameras and scanners. In Elizabeth Allen and Sophie Triantaphillidou, editors, *The Manual of Photography (Tenth Edition)*, pages 263–288. Focal Press, Oxford, tenth edition edition, 2011.

[24] Stemmer Imaging. *The Imaging & Vision Handbook*. Stemmer imaging AG, 2018.

[25] Sidney Ray. Chapter 10 - camera lenses. In Elizabeth Allen and Sophie Triantaphillidou, editors, *The Manual of Photography (Tenth Edition)*, pages 175–197. Focal Press, Oxford, tenth edition edition, 2011.

[26] Elizabeth Allen. Chapter 1 - introduction to the imaging process. In Elizabeth Allen and Sophie Triantaphillidou, editors, *The Manual of Photography (Tenth Edition)*, pages 1–18. Focal Press, Oxford, tenth edition edition, 2011.

[27] Light basics. `https://www.opto-e.com/basics/light-basics`.

[28] Illumination geometries and techniques. `https://www.opto-e.com/basics/illumination-geometries-and-techniques`.

[29] Bright field, front light illumination. `https://www.opto-e.com/basics/bright-field-front-light-illumination`.

[30] Dark field, backlight illumination. https://www.opto-e.com/basics/dark-field-backlight-illumination.

[31] Shuai Wang and Zhendong Su. Metamorphic testing for object detection systems, 2019.

[32] Farhana Sultana, Abu Sufian, and Paramartha Dutta. Evolution of image segmentation using deep convolutional neural network: a survey. *Knowledge-Based Systems*, 201:106062, 2020.

[33] Mohammad Hossin and Md Nasir Sulaiman. A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5(2):1, 2015.

[34] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[35] Paul Jaccard. The distribution of the flora in the alpine zone.1. *New Phytologist*, 11(2):37–50, 1912.

[36] Phil Kim. *Convolutional Neural Network*, pages 121–147. Apress, Berkeley, CA, 2017.

[37] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[38] Sebastian Ruder. An overview of gradient descent optimization algorithms, 2016.

[39] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. Ieee, 2017.

[40] Comprehensive guide to different pooling layers in deep learning. https://analyticsindiamag.com/comprehensive-guide-to-different-pooling-layers-in-deep-learning/, Aug 2021.

[41] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[42] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[43] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[44] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2017.

[45] Freepik. https://www.flaticon.com/authors/freepik. Images: End mill, camera, computer, calipers, and database.

[46] Smashicons. https://www.flaticon.com/free-icons/cnc-machine. Image: Cnc machine.

[47] International Standard Organization. Tool life testing in milling—part 2: end milling, 1989.

[48] Line scan applications. https://www.stemmer-imaging.com/en-se/knowledge-base/cameras-line-scan-applications/.

[49] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[50] Arun Gandhi. Data augmentation: How to use deep learning when you have limited data. https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/, May 2021.

[51] María Teresa García-Ordás, Enrique Alegre-Gutiérrez, Víctor González-Castro, and Rocío Alaiz-Rodríguez. Combining shape and contour features to improve tool wear monitoring in milling processes. *International Journal of Production Research*, 56(11):3901–3913, 2018.

[52] Basler racer: User's manual for gige vision cameras, Apr 2019.

[53] Laowa 100mm f/2.8 2x ultra macro apo - laowa camera lenses. https://www.venuslens.net/product/laowa-100mm-f-2-8-2x-macro-apo/, Jan 2022.

[54] Dark field, backlight illumination. https://latab.net/fileadmin/user_uploads_LATAB/Products/Datenblaetter/LATAB_LineLights_square_2018.pdf, 2014.

[55] Servo motor emme-as-100-m-hs-amb. https://www.festo.com/us/en/a/download-document/datasheet/2103502, Mar 2022.

[56] Servo motor emmt-as-60-m-ls-rmb. https://www.festo.com/us/en/a/download-document/datasheet/5242207, Mar 2022.

[57] A. Dutta, A. Gupta, and A. Zissermann. VGG image annotator (VIA). http://www.robots.ox.ac.uk/~vgg/software/via/, 2016. Version: 2.0.11, Accessed: 2022-03-16.

[58] Source code for torchvision models detection mask r-cnn. https://pytorch.org/vision/stable/_modules/torchvision/models/detection/mask_rcnn.html. Accessed: 2022-04-12.

[59] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[60] Petro Liashchynskyi and Pavlo Liashchynskyi. Grid search, random search, genetic algorithm: A big comparison for nas, 2019.

[61] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In Věra Kůrková, Yannis Manolopoulos, Barbara Hammer, Lazaros Iliadis, and Ilias Maglogiannis, editors, *Artificial Neural Networks and Machine Learning – ICANN 2018*, pages 270–279, Cham, 2018. Springer International Publishing.

[62] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing.

[63] Transforming and augmenting images. https://pytorch.org/vision/stable/transforms.html. Accessed: 2022-04-30.

A

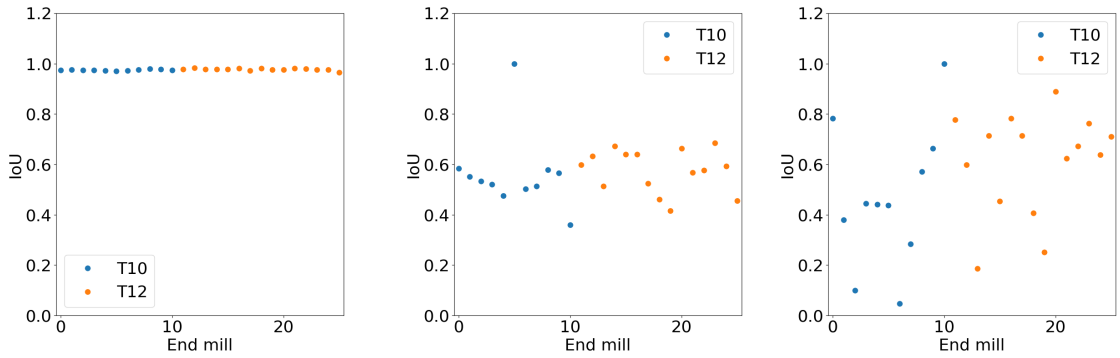
Results From All Models

Result for all WSNs.

A.1 Segmentation

A.1.1 T10 and T12 Tools

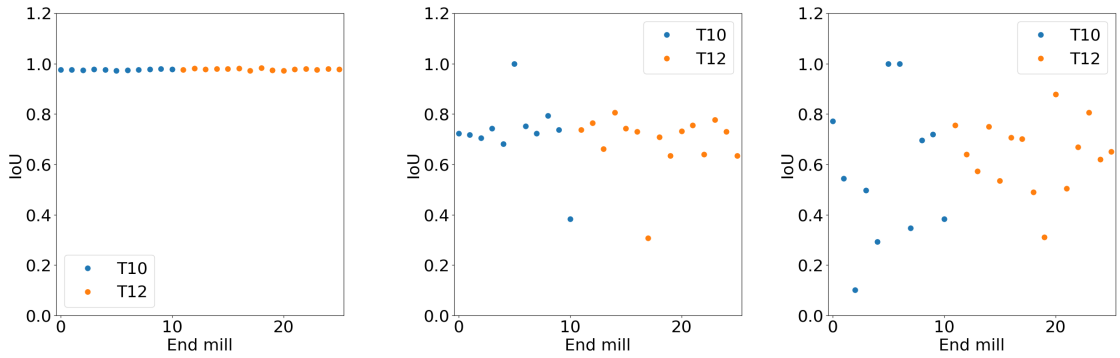
WSN Base



(a) Flank face. **(b)** Flank Wear. **(c)** Fracture.

Figure A.1: IoU distribution for WSN Base on images of the T10 and T12 tools in the testing data set

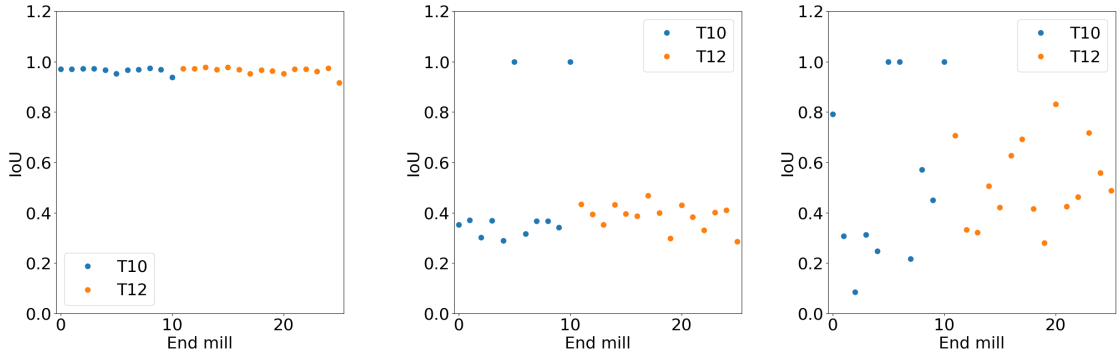
WSN Rot



(a) Flank face. **(b)** Flank Wear. **(c)** Fracture.

Figure A.2: IoU distribution for WSN Rot on images of the T10 and T12 tools in the testing data set

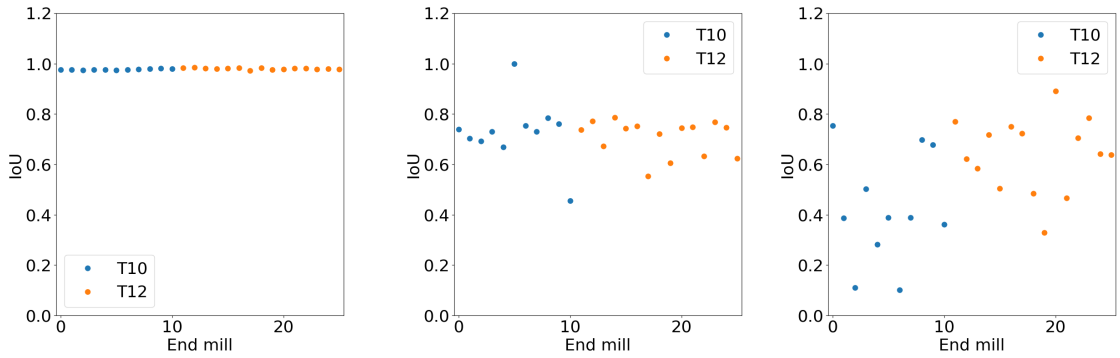
WSN Aug



(a) Flank face. **(b)** Flank Wear. **(c)** Fracture.

Figure A.3: IoU distribution for WSN Aug on images of the T10 and T12 tools in the testing data set

WSN Mask20

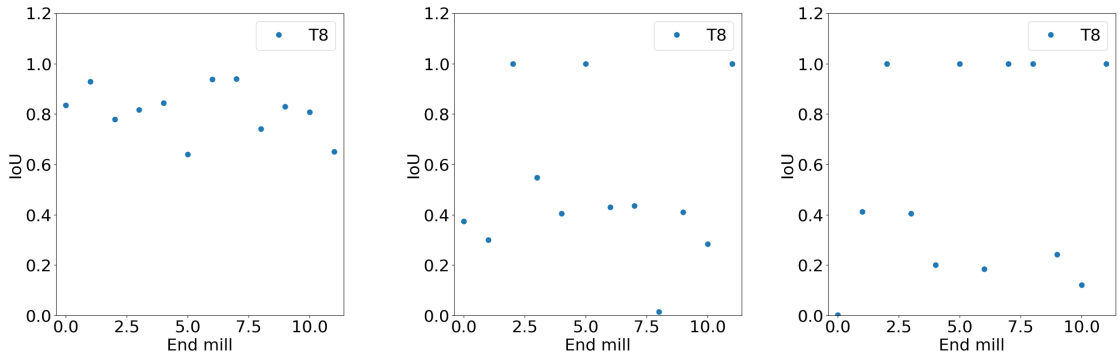


(a) Flank face. **(b)** Flank Wear. **(c)** Fracture.

Figure A.4: IoU distribution for WSN Mask20 on images of the T10 and T12 tools in the testing data set

A.1.2 T8 Tools

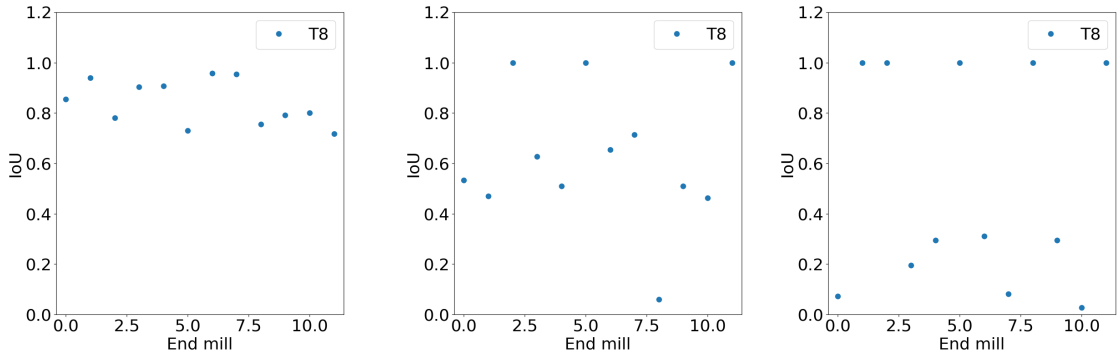
WSN Base



(a) Flank face. **(b)** Flank Wear. **(c)** Fracture.

Figure A.5: IoU distribution for WSN Base on images of the T8 tools in the testing data set

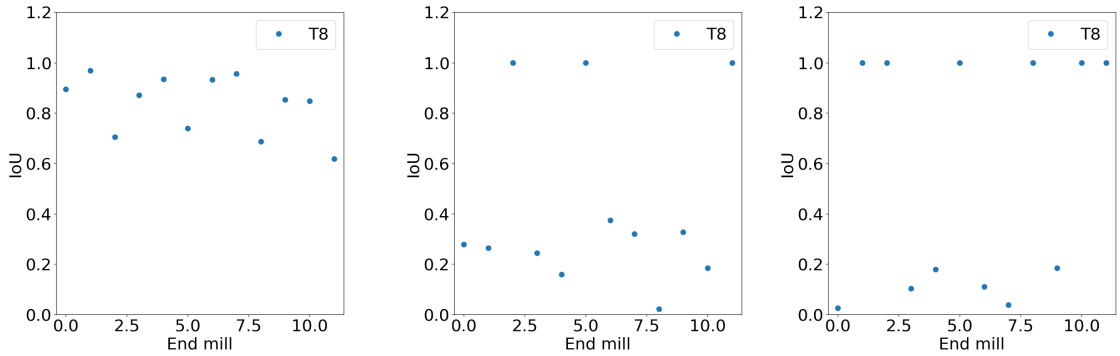
WSN Rot



(a) Flank face. **(b)** Flank Wear. **(c)** Fracture.

Figure A.6: IoU distribution for WSN Rot on images of the T8 tools in the testing data set

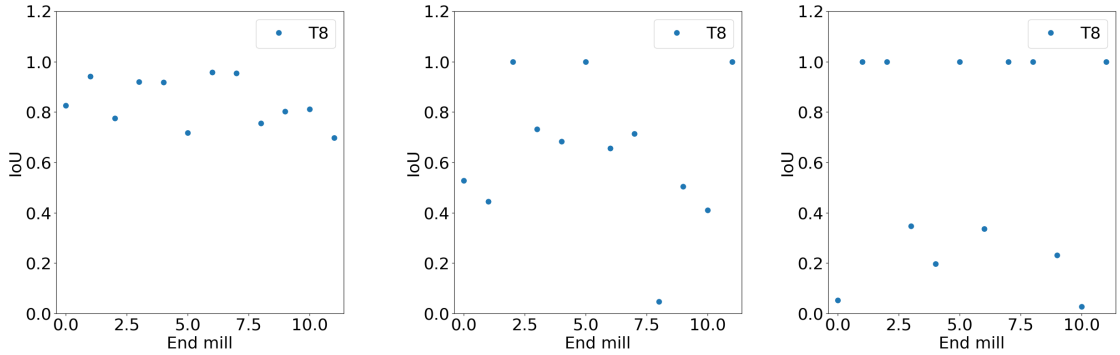
WSN Aug



(a) Flank face. **(b)** Flank Wear. **(c)** Fracture.

Figure A.7: IoU distribution for WSN Aug on images of the T8 tools in the testing data set

WSN Mask20



(a) Flank face. **(b)** Flank Wear. **(c)** Fracture.

Figure A.8: IoU distribution for WSN Mask20 on images of the T8 tools in the testing data set

A.2 Quantification

A.2.1 T10 and T12 Tools

WSN Base

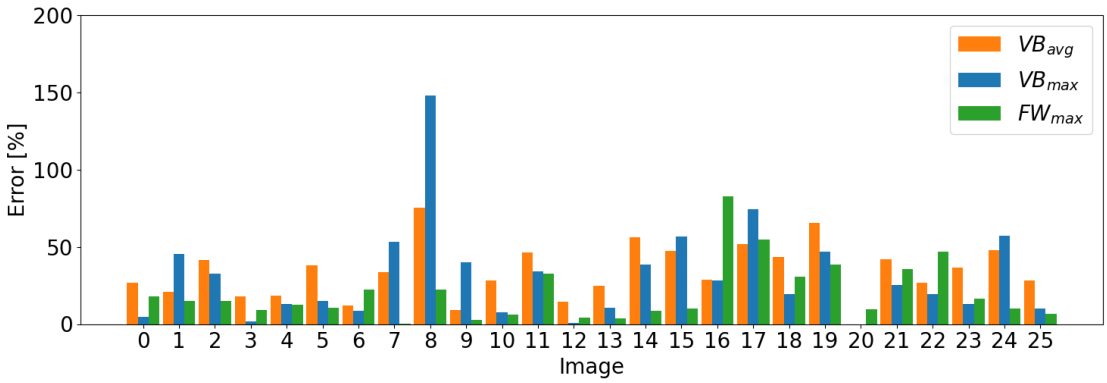


Figure A.9: Distribution of the relative absolute error for quantification of T10 and T12 tools in the testing data set. The quantification is done with masks generated by WSN Base. If the error is shown to be zeros it is because that wear type did not exists in the image (Thus it does not indicate a perfect measurement).

WSN Rot

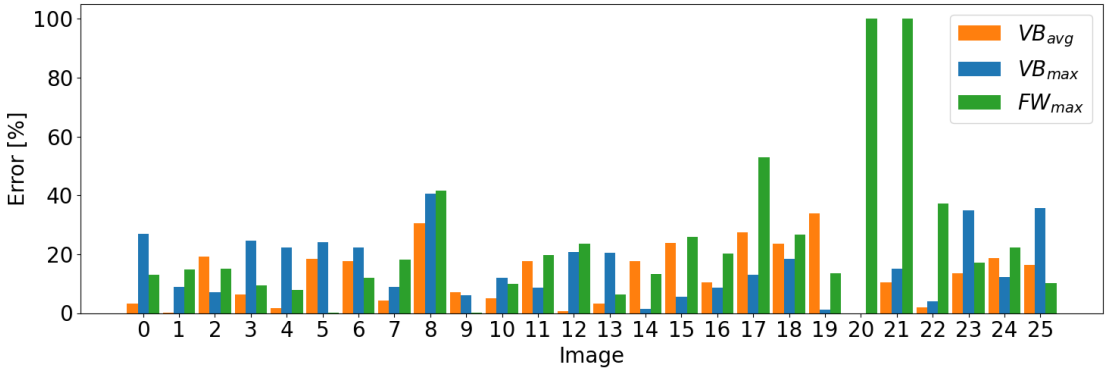


Figure A.10: Distribution of the relative absolute error for quantification of T10 and T12 tools in the testing data set. The quantification is done with masks generated by WSN Rot. If the error is shown to be zeros it is because that wear type did not exists in the image (Thus it does not indicate a perfect measurement).

WSN Aug

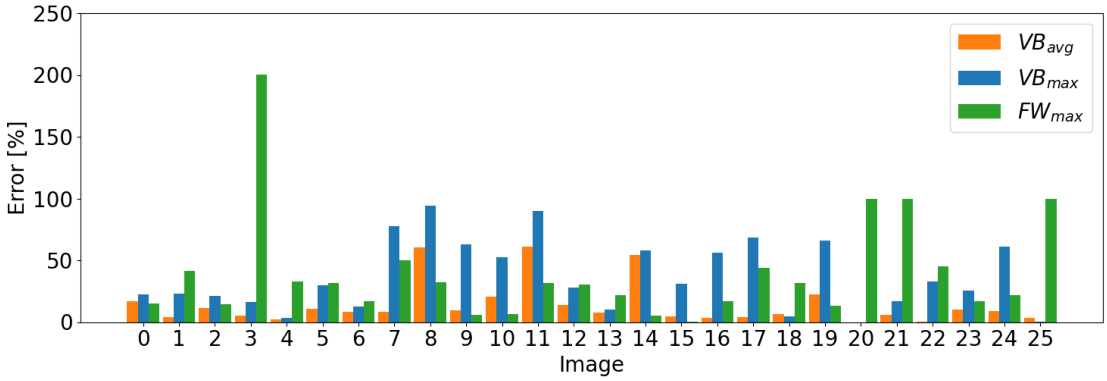


Figure A.11: Distribution of the relative absolute error for quantification of T10 and T12 tools in the testing data set. The quantification is done with masks generated by WSN Aug. If the error is shown to be zeros it is because that wear type did not exists in the image (Thus it does not indicate a perfect measurement).

WSN Mask20

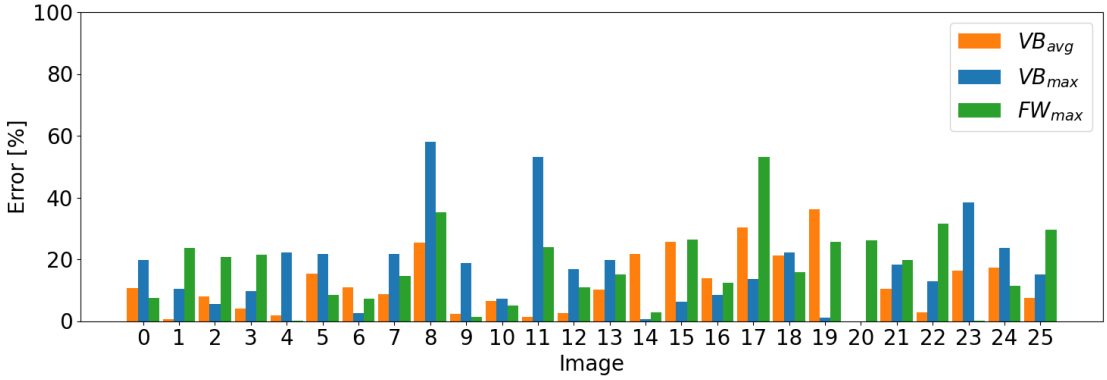


Figure A.12: Distribution of the relative absolute error for quantification of T10 and T12 tools in the testing data set. The quantification is done with masks generated by WSN Mask20. If the error is shown to be zeros it is because that wear type did not exists in the image (Thus it does not indicate a perfect measurement).

A.2.2 T8 Tools

WSN Base

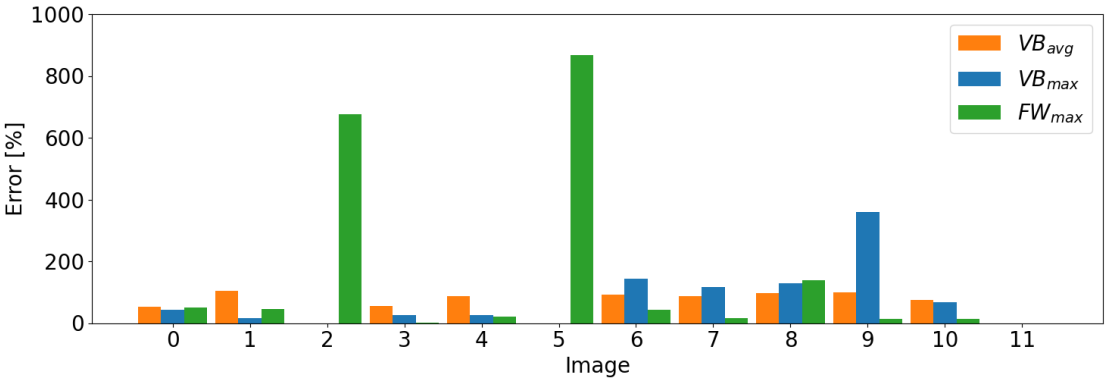


Figure A.13: Distribution of the relative absolute error for quantification of T8 tools in the testing data set. The quantification is done with masks generated by WSN Base. If the error is shown to be zeros it is because that wear type did not exists in the image (Thus it does not indicate a perfect measurement).

WSN Rot

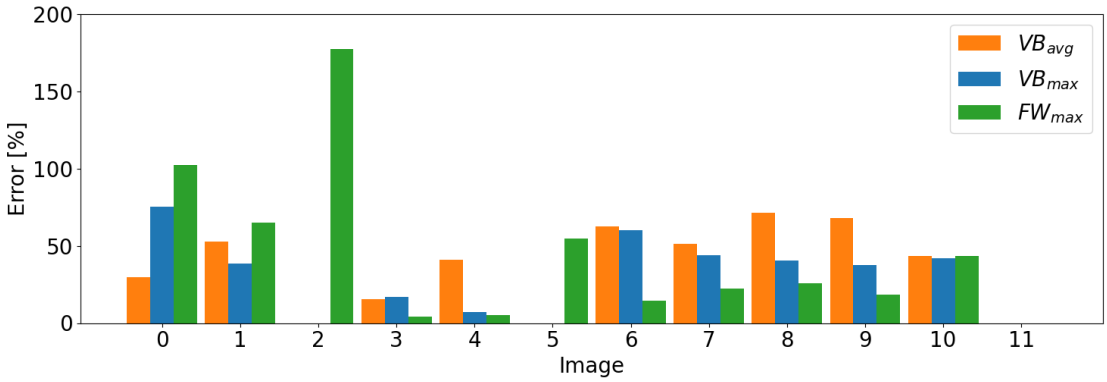


Figure A.14: Distribution of the relative absolute error for quantification of T8 tools in the testing data set. The quantification is done with masks generated by WSN Rot. If the error is shown to be zeros it is because that wear type did not exists in the image (Thus it does not indicate a perfect measurement).

WSN Aug

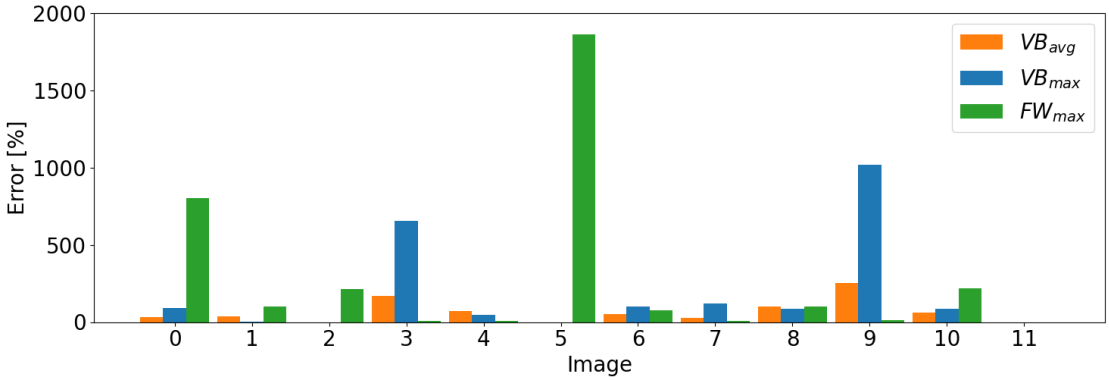


Figure A.15: Distribution of the relative absolute error for quantification of T8 tools in the testing data set. The quantification is done with masks generated by WSN Aug. If the error is shown to be zeros it is because that wear type did not exists in the image (Thus it does not indicate a perfect measurement).

WSN Mask20

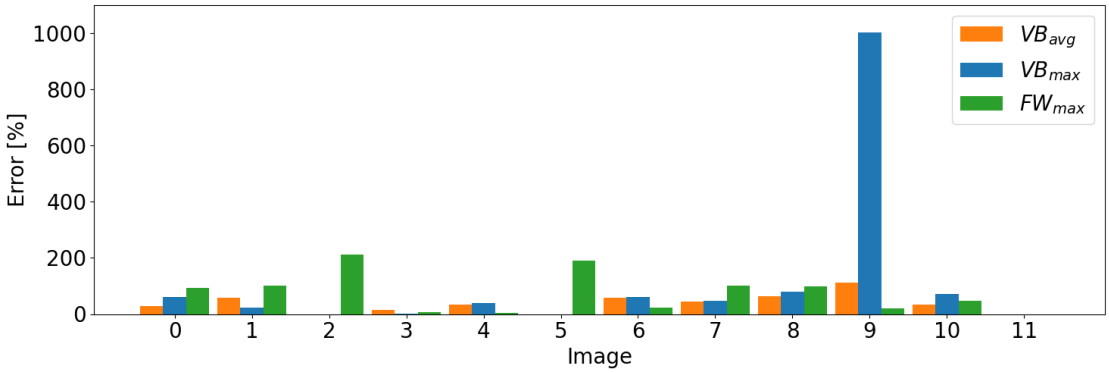


Figure A.16: Distribution of the relative absolute error for quantification of T8 tools in the testing data set. The quantification is done with masks generated by WSN Mask20. If the error is shown to be zeros it is because that wear type did not exists in the image (Thus it does not indicate a perfect measurement).

B

Python Environment

The python environment was managed with anaconda version 4.10.3.

B.1 Setup

The environment was created by running the following commands in the Anaconda Prompt:

```
conda create -n maskRCNN python=3.7
activate maskRCNN
conda install pytorch torchvision cudatoolkit=11.3 -c pytorch
pip install tenosrboard
pip install scikit-image
pip install opencv-python
pip install pycocotools-windows
```

B.2 Run

A program was run by first activating the environment and then running the script, with the two following commands:

```
activate maskRCNN
python [python file to run]
```