LAPPEENRANTA UNIVERSITY OF TECHNOLOGY
FACULTY OF TECHNOLOGY MANAGEMENT
INFORMATION TECHNOLOGY

VISUAL MEASUREMENT AND MODELLING OF TOOL WEAR IN ROUGH TURNING

Master's thesis

Supervisor: Adjunct Professor Ville Kyrki

Examiners: Adjunct Professor Ville Kyrki and Professor Juha Varis

Lappeenranta, February 26, 2008

Janne Laaksonen
Punkkerikatu 5 B 23
53850 Lappeenranta
Tel. +358-40-5405888
Janne.Laaksonen@lut.fi

ABSTRACT

Lappeenranta University of Technology

Faculty of Technology Management

Information Technology

Janne Laaksonen

Visual Measurement and Modelling of Tool Wear in Rough Turning

Master's thesis

2008

98 pages, 54 figures, 13 tables and 5 appendices.

Examiners: Adjunct Professor Ville Kyrki and Professor Juha Varis

Keywords: Machine Vision, Rough Turning, Measurement of Flank Wear, Modelling of Flank Wear

Despite the high automation level in turning industry, a few key issues prevent complete automation of the entire turning process. One of these issues is tool wear. The work presented in the thesis focuses on implementing an automatic system to measure tool wear, especially flank wear, using machine vision. The tool wear measurement system removes the need for manual inspection and minimizes the time used on wear measurement. In addition to measurement, tool wear modelling and prediction is also studied.

The automatic measurement system was placed inside a turning center and the system was successfully integrated with external systems. Conducted experiments showed that the measurement system is able to measure the tool wear successfully in the system's intended environment. The measurement system can also tolerate some common disturbances affecting machine vision systems.

Multiple methods that can model the tool wear were studied, including neural networks and support vector regression. Experiments showed that the models were able to predict the wear based on the time the tool has been in use. Best results were obtained by neural networks with Bayesian regularization.

TIIVISTELMÄ

Lappeenrannan teknillinen yliopisto

Teknistaloudellinen tiedekunta

Tietotekniikka

Janne Laaksonen

Visual Measurement and Modelling of Tool Wear in Rough Turning

Diplomityö

2008

98 sivua, 54 kuvaa, 13 taulukkoa ja 5 liitettä.

Tarkastajat: Dosentti Ville Kyrki ja professori Juha Varis

Hakusanat: Konenäkö, Rouhintasorvaus, Viistekulumisen mittaus, Viistekulumisen mallinnus

Huolimatta korkeasta automaatioasteesta sorvausteollisuudessa, muutama keskeinen ongelma estää sorvauksen täydellisen automatisoinnin. Yksi näistä ongelmista on työkalun kuluminen. Tämä työ keskittyy toteuttamaan automaattisen järjestelmän kulumisen, erityisesti viistekulumisen, mittaukseen konenäön avulla. Kulumisen mittausjärjestelmä poistaa manuaalisen mittauksen tarpeen ja minimoi ajan, joka käytetään työkalun kulumisen mittaukseen. Mittauksen lisäksi tutkitaan kulumisen mallinnusta sekä ennustamista.

Automaattinen mittausjärjestelmä sijoitettiin sorvin sisälle ja järjestelmä integroitiin onnistuneesti ulkopuolisten järjestelmien kanssa. Tehdyt kokeet osoittivat, että mittausjärjestelmä kykenee mittaamaan työkalun kulumisen järjestelmän oikeassa ympäristössä. Mittausjärjestelmä pystyy myös kestämään häiriöitä, jotka ovat konenäköjärjestelmille yleisiä.

Työkalun kulumista mallinnusta tutkittiin useilla eri menetelmillä. Näihin kuuluivat muiden muassa neuroverkot ja tukivektoriregressio. Kokeet osoittivat, että tutkitut mallit pystyivät ennustamaan työkalun kulumisasteen käytetyn ajan perusteella. Parhaan tuloksen antoivat neuroverkot Bayesiläisellä regularisoinnilla.

Contents

1 INTRODUCTION **1**

1.1 Background 1

1.2 Objectives and Restrictions 1

1.3 Structure of the Thesis 2

2 TOOL WEAR MEASUREMENT AND PREDICTION **3**

2.1 Cutting Tool Wear and Measurement of Tool Wear 3

2.2 Machine Vision in Tool Wear Measurement 7

2.2.1 Measuring tool wear directly 8

2.2.2 Tool wear estimation indirectly 10

2.3 Tool Wear Prediction and Modeling 10

3 REQUIREMENTS OF THE VISUAL MEASUREMENT SYSTEM **15**

3.1 Hardware 15

3.2 Software 17

4 MEASURING TOOL WEAR WITH MACHINE VISION **19**

4.1 Machine Vision 19

4.2 Algorithm for Measuring Flank Wear 21

4.2.1 Algorithm 21

4.2.2 Bresenham's line algorithm 28

4.2.3 Canny edge detector 29

4.2.4 Hough transform 30

4.2.5 Considerations 33

5 MODEL BASED PREDICTION OF TOOL WEAR **34**

5.1 Modeling Flank Wear 34

5.2 Flank Wear Prediction With A Differential Model 35

5.3 Linear Regression 36

5.4 Artificial Neural Networks 37

5.5 Support Vector Regression 39

6 VISUAL MEASUREMENT SYSTEM IMPLEMENTATION **40**

6.1 Hardware components 40

6.1.1 Turning center 40

6.1.2 Imaging System 42

6.1.3 Communication 44

6.1.4 Considerations 45
6.2 Software components 47
6.2.1 Software prototype and libraries 47
6.2.2 Overview of the software system 48
6.2.3 Software structure 48
6.2.4 Software functionality 51
6.2.5 Communication 52
6.2.6 Documentation 53
6.2.7 Parameter settings 53

7 EXPERIMENTS AND DISCUSSION 56
7.1 Visual measurement system 56
7.1.1 Synthetic tests 56
7.1.2 Application tests 63
7.2 Modelling experiments 68
7.2.1 Wear experiments 69
7.2.2 Linear regression 72
7.2.3 Artificial neural networks 72
7.2.4 Support vector regression 73
7.2.5 Comparison of modeling methods as predictors of wear 74

8 CONCLUSIONS 76

REFERENCES 78

APPENDIX 1. System Specification

APPENDIX 2. Specifications for camera, lens and lighting

APPENDIX 3. Algorithms

APPENDIX 4. Software class diagram

APPENDIX 5. Parameter List

ABBREVIATIONS AND SYMBOLS

b	Depth of a cut
f	Cutting feed
r_ϵ	Tool nose radius
v	Cutting speed

α	angle
ρ	Polar coordinate distance
σ	Deviation of a Gaussian distribution
θ	Polar coordinate angle

ANN	Artificial Neural Network, a type of learning algorithm
CCD	Charge Coupled Device, a device used to capture images
CID	Charge Injection Device, a device used to capture images
CNC	Computer Numerical Control
DOE	Design of Experiments, a method to efficiently design experiments
ISO	International Organization for Standardization
LED	Light-Emitting Diode
MSC	Message Sequence Chart, a diagram type to represent communication
RAM	Random Access Memory, a type of memory used in computers
SVM	Support Vector Machine, a type of learning algorithm
SVR	Support Vector Regression, a data fitting method based on SVM
TCP/IP	Transmission Control Protocol over Internet Protocol, communication protocol
UML	Unified Modeling Language, a language to represent software

1 INTRODUCTION

The goal of this section is to provide a background and motivation for this thesis. The structure of the thesis is also given.

1.1 Background

Modern turning in industrial setting is done by CNC (Computer Numerical Control) lathes. These CNC lathes allow programmability and high automation regarding the turning process. There is, however, one critical component, which is still done by hand: measuring the wear of the cutting tool used in turning. There is no any industry wide adoption for automating the process of measuring the wear. Instead wear measurement requires stopping the automated turning, removing the tool, measuring the tool and putting the tool back into it's holder. The time lost for measurement that the tool will only last minutes in normal turning.

This thesis is linked to the Feedchip-project (2006-2008), which tries to maximize the production rate in turning. As a part of the Feedchip-project, the work done for this thesis tries to minimize the time spent on measuring the wear on the cutting tool by applying machine vision and prediction of the tool wear to the problem.

1.2 Objectives and Restrictions

This thesis has two objectives. The first is to show that it is possible to use machine vision to automate measuring of flank wear. Flank wear was chosen as the measured variable as it was identified as the most important wear type in the Feedchip-project. No other form of wear is measured. Only a single camera is used to capture images of the tool. Automated measuring system is integrated to an external control system.

The second objective is to develop a method for predicting the cutting tool life based on the cutting conditions and additional sensor information collected during turning. This can be used in optimizing the cutting tool changes which leads to maximized production. Developing a prediction method, which can work under changing cutting conditions during turning, was identified as especially important. In this thesis only one combination of

cutting tool and work material is considered for modelling. Contrary to measuring, the modelling will be examined on a more theoretical level, with no actual implementation or integration to the turning process.

1.3 Structure of the Thesis

The rest of this thesis is divided into seven sections. Section 2 includes a literature review on the topic of tool condition monitoring and on the prediction and estimation of tool wear. The Section also includes an introduction to the cutting tools used in turning and their monitoring.

Section 3 gives the design point of view to the system. The section describes how the system should work and what it must do.

Sections 4 and 5 contain the theory of the methods used in the practical part of the thesis. Section 4 focuses on the tool wear measurement and machine vision while Section 5 describes the theory behind the prediction and estimation of tool wear. Section 4 also describes general machine vision systems in the context of this thesis.

Section 6 is a description of the implementation of the machine vision system. It includes both the software and hardware used in the system.

Section 7 describes the conducted experiments and discusses their results. Finally, conclusions in Section 8 mark the ending of the thesis.

2 TOOL WEAR MEASUREMENT AND PREDICTION

Several different methods have been developed to monitor the condition of tools in turning. This thesis focuses specifically on cutting tools, also known as inserts, used in turning. All tool wear monitoring methods related to the thesis' subject are briefly discussed here, while existing machine vision methods are explained in detail in Section 2.2. Also a brief introduction to measuring and classifying the condition of a cutting tool is given in Section 2.1.

The prediction of tool wear has also had interest in the last few years. The prediction of the tool wear allows optimizing the tool changes as the tool life is known before the tool is actually rejected. There are multiple ways to predict the tool life and these will be discussed in Section 2.3.

2.1 Cutting Tool Wear and Measurement of Tool Wear

ISO standard 3685:1993 [1] is the standard for measuring the wear for wear experiments when using a single-point turning tool. The word "single-point" refers to the fact that the tool cuts the material with a single point. Figure 1 depicts a cutting tool.

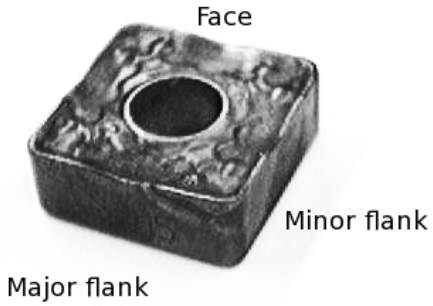


Figure 1: Image of a cutting tool

From the Figure 1, it can be seen that the tool consists of three important sides: the major flank, the minor flank and the face at the top. However, the ordering of the sides is dependent on the application. Major flank is the cutting edge, while minor flank faces the newly cut surface and the face receives the material being cut and forms chips. The type of cutting tool presented in Figure 1 has four usable cutting edges, one on each side

of the tool. Of these, the most useable for measuring are the major flank and the face as the standard [1] gives threshold values for wear experiments on the wear types occurring on the two sides.

The standard [1] also defines four definitions which will be used in this thesis as they are defined in the standard:

Tool wear: The change of shape of the tool from its original shape, during cutting, resulting from the gradual loss of tool material or deformation.

Tool wear measure: A dimension to be measured to indicate the amount of tool wear.

Tool-life criterion: A predetermined threshold value of a tool wear measure or the occurrence of a phenomenon.

Tool life: The cutting time required to reach a tool-life criterion.

According to the ISO standard 3685:1993, there are multiple types of wear and phenomena, which can cause tool-life criterion to be fulfilled. Most important of the wear types are flank wear and crater wear. Flank wear is present in all situations and it is the best known type of wear. It can be found on the major flank of the tool. Crater wear appears on the face of the tool as a crater. Crater wear is the most commonly found wear on the face of the tool.

The wear process itself changes under the influence of different conditions. However, three main factors contributing to the wear are known: adhesion, abrasion and diffusion. Adhesion occurs when the work material, that the tool is cutting, welds onto the tool. This happens because of the friction between the tool and work material, which generates heat. When these welds are broken, small pieces of the tool are lost. Abrasion is mechanical wear resulting from the cutting action, where the tool grinds itself on to the work material. Diffusion wear occurs on a narrow reaction zone between the tool and work material. In diffusion wear the atoms from the tool move to the work material. This usually accelerates the other two wear processes as the tool material is weakened. [2, pp. 141-142]

Figure 2 shows the general curves of flank wear. The wear is typically characterized by rapid initial wear, a linear increase of the wear in the middle of the tool life and finally a rapid increase of the wear rate before the tool breaks completely. [2, p. 143]

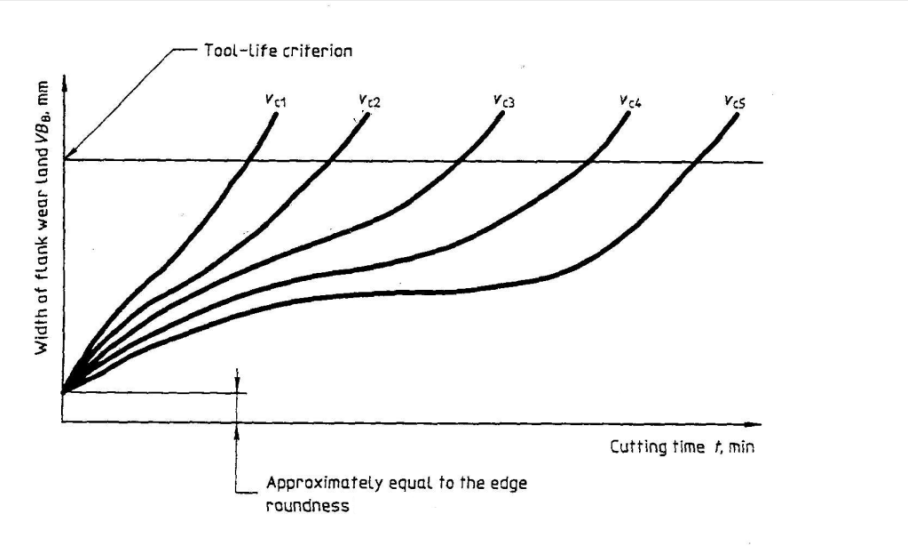


Figure 2: Flank wear progress as a function of time [1]

While the general shape of the curve stays the same, cutting conditions or cutting parameters affect the tool life, *i.e.* the gradient of the curve, especially the linear section. Most important cutting parameters in relation to tool wear are cutting speed, denoted by V, and cutting feed, denoted by f. Of these speed is considered to have the most effect on the tool life. However, Astakhov [3] argues that there are conflicting results and that feed has significant effect under optimal cutting temperature. The importance of cutting speed can also be seen from the Taylor's equation as the formula relies only on the cutting speed to estimate tool life. Taylor's equation is described in Section 2.3. [4, pp. 60-62]

The effects of cutting speed can be seen from Figure 2. The Cutting speed is usually expressed in meters per minute and can range up to $500\ m/min$ [4]. The standard [1] gives a limit to the cutting speeds by limiting the minimum tool life to 5 minutes or to 2 minutes when using a ceramic tool. The cutting feed is expressed as millimeters per revolution, ranging from 0.1 to $0.9\ mm/rev$ [4, p. 61]. Standard [1] has defined the maximum feed to 0.8 times the corner radius of the tool. The corner radius can be seen in Figure 3.

To be able to measure the wear, especially flank wear, a few variables must be known. Most important variables are r_ϵ, corner radius, and b, depth of cut. These are represented in Figure 3. This figure is depicting the tip or nose of the tool viewing it from the top. The arrow in the figure represents the cutting direction.

The variable r_ϵ tells the corner radius, *i.e.* how round the tip of the tool is. b on the other

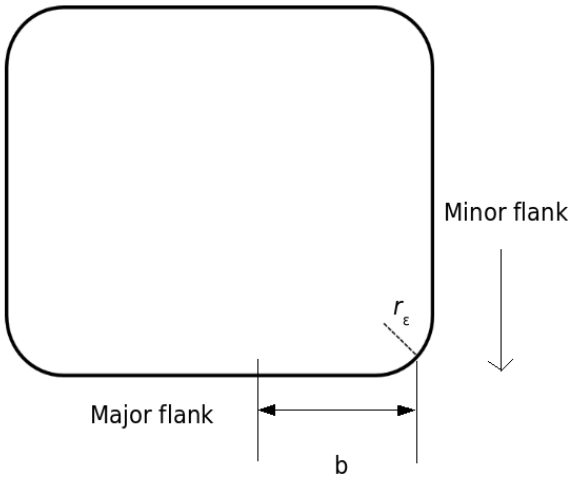


Figure 3: Nose of the tool

hand tells the depth of the cut used to cut the work-piece. These are used in measuring the flank wear. The measurements defined for flank wear in the standard [1] are shown in Figure 4. The figure depicts the major flank of the tool from the side and the tip of the tool lies at the top.

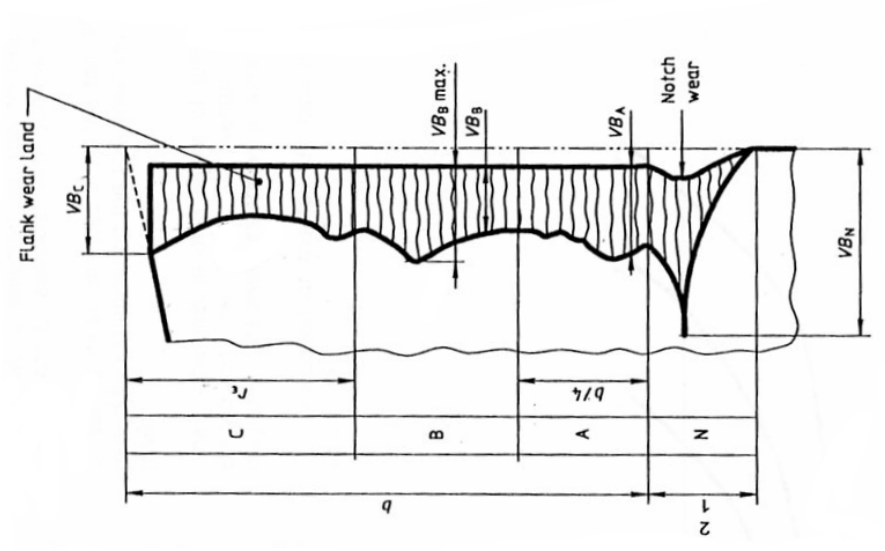


Figure 4: Measurement of flank wear [1]

In Figure 4, two measures are the most important, as they are designated in the standard [1] to be tool-life criterions. These measures are VB_B and VB_B max. Besides these most important measures, also VB_C is measured, which is considered to be the maximum wear width in zone C. VB_B and VB_B max. govern the tool-life by following criteria:

a) the maximum width of the flank wear land VB_B max. = 0.6 mm if the flank wear land is not regularly worn in zone B.

b) the average width of the flank wear land VB_B = 0.3 mm if the flank wear land is considered regularly worn in zone B.

Commonly the measurements have been done by hand using a microscope. In the standard [1] manual measurement is defined as the way to measure the wear. So to follow the standard, measurements done with an automatic system need to be close to the manual measurements.

Kurada *et al.* [5] divide the available methods for tool condition monitoring into two parts: direct and indirect sensors. Direct sensors are sensors which measure the tool wear directly. These sensors include vision, proximity and radioactive sensors. Proximity sensors estimate the wear by measuring the distance between the workpiece and tool edge. Radioactive sensors measure the amount of radioactive material transferred from the tool, which has been infused with radioactive material, to the chips of the work-piece. Vision sensors will be dealt separately in Section 2.2 as they are in the scope of this thesis. Direct measurements can usually be taken only between the machining runs, because the major flank of the tool is not visible during actual machining. [5]

Indirect sensors can only estimate the wear as they have no direct way to measure the actual wear, but instead they rely on secondary effects of the wear, such as increase in cutting forces. To indirectly measure the wear, measurements of cutting force, vibration and acoustic emission during the machining have been proposed. Also surface texture of the work-piece has been used to estimate the wear of the tool. These methods will also be explained in Section 2.2 as the texture can be captured using machine vision. Indirect measurements have the advantage of being on-line measurements, *i.e.* they can be measured during machining. [5]

2.2 Machine Vision in Tool Wear Measurement

Machine vision systems have been developed to measure the wear of the tool both directly and indirectly. Direct measurement systems have been shown to be successful in determining the wear, flank or otherwise, from images taken from the tool [6], [7], [8]. However, most of the systems do not operate inside the CNC lathe or during the ma-

chining process, or in-cycle [6], [7], which diminishes their usefulness in real industrial applications. Machine vision systems utilizing indirect measurements have been focusing on the texture of the machined work-piece surface [9], [10]. The roughness of the surface can be used as a quality measure.

As this field of applied machine vision is quite specific, only a few comprehensive reviews of methods have been made. Kurada *et al.* [5] gives a round-up of contemporary methods. While Rehorn *et al.* [11] give a broad view of tool condition monitoring methods, including other indirect measuring methods, some machine vision methods are reviewed as well. Methods represented in sections 2.2.1 and 2.2.2 have been studied with the use of turning unless otherwise noted.

2.2.1 Measuring tool wear directly

Kurada *et al.* [7] designed a machine vision system that can measure flank wear. Main components of the system were a CCD camera with a zoom lens and two fiber optic illuminators. The resolution in the captured image was approximately $3\mu m$ per pixel. Images are captured orthogonally to the flank face. Actual wear region identification and measurement relies on the specular reflection of the wear area from the fiber optic lights. Image segmentation is based on texture. Variance operator is used to convert areas of similar texture to similar intensity values. After this, the image is thresholded to bring out the wear area. To accommodate noise in the image, connected areas are extracted. Largest of these areas is determined to be the wear region. Measurements are calculated from this area.

Kurada *et al.* [7] tried also other methods to extract the wear region, such as edge detectors and other texture operations, but came to a conclusion that variance operator was best suited for detecting the wear. Results were promising as the wear measurements were repeatable and comparable to manual measurements, but the system itself was not integrated to a turning machine.

Lanzetta [8] has a different approach to the problem. Using two cameras, a grey-scale image is taken from the nose of the tool and image of the tool silhouette is taken from the top of the tool. Silhouette is taken with a back light and the tool nose is directly illuminated. Images are also taken with directional light from the top of the tool to detect crater wear. Flank wear is measured from the silhouette image. The silhouette of the worn

tool is compared to the silhouette of the unused tool, which gives an estimate of the tool flank wear. Measurement error is reported to be less than 5 %.

Prasad *et al.* [12] measure the crater wear using stereo-vision. Stereo-vision is not achieved by using two cameras but with the tool moving between two images taken with the same grey-scale camera. Resolution is $12\ \mu m \times 10\ \mu m$ per pixel. The depth and width of the crater wear is measured and the measurements can be made accurately. However the images of the tool are taken outside the turning machine and no method for automatically measuring the flank wear is presented.

Jurkovic *et al.* [13] used structured light from a laser diode to capture elevation information from the tool surface. A monochrome CCD camera is used to capture the images with a halogen light. The resolution of the image was approximately $3\mu m$ per pixel. Flank wear can be measured as well as crater wear using the elevation information. The system was combined with a CNC lathe.

Kerr *et al.* [14] used a monochrome CCD camera to capture images from the tool nose. Variety of methods were tested to compare their results in extracting the wear information from the tool. Methods included edge operators, texture information, histogram analysis, Fourier transform and fractal properties of the image. Texture information was found to be the most useful and accurate in measuring the extent of the wear. This is the same conclusion as in [7]. However, the system was not implemented to be used in-cycle, the tool had to be removed from the tool holder.

Park *et al.* [6] measure flank wear of tools using charge injection device (CID) camera with $36\ \mu m \times 46\ \mu m$ resolution. The measurement is based on specular light reflecting from the worn area. Based on this reflection the image is segmented to worn and unworn areas. Accurate measurement results are reported.

A few common features can be seen from these studies. One is the reliance on the specular reflection of the wear area as reported on three studies. The worn area is usually polished from the grinding effect of turning and it has a different angle than the rest of the major flank. This allows the specular reflection to occur only at the worn area. Second feature is the use of texture operators which helps to segment the image in worn and unworn areas. Most of the studies presented here, which measure flank wear, only measure one of the VB_B, VB_B max. or VB_C measures.

2.2.2 Tool wear estimation indirectly

Kassim *et al.* [9] study the surface of the work-piece to determine the wear of the tool. Two methods are explored, column-projection and Hough transform. Column-projection is found suitable for detecting wear levels of the tool in turning. However, only rough estimates can be made of the tool wear and as the result, estimates are categorized into three categories: sharp, semi-dull and dull.

Bradley *et al.* [10] study the use of the image histogram, frequency transforms and texture-based segmentation to estimate the wear on the tool using the surface texture of the work-piece. This study was conducted with the use of face milling instead of turning. Bradley *et al.* come to the conclusion that using frequency domain analysis and texture-based analysis, the tool wear can be estimated.

Using vision to estimate tool wear indirectly is concentrated on studying the surface quality of the work-piece. There are no other sources for imaging that can be used to estimate tool wear, which is why the surface quality is used. It is also possible to evaluate the actual surface quality in addition to the tool wear.

2.3 Tool Wear Prediction and Modeling

ISO standard 3685:1993 [1] defines a way, Taylor's basic equation, to calculate tool life from experimental data. Experimental data is gathered by measuring the time until the tool-life criterion is fulfilled using several different cutting speeds. From several of these experiments a curve representing the tool life can be constructed. An example of a tool life curve is shown in Figure 5 with two criterions used for

The standard [1] allows the curve to be fitted to the data by eye or by statistical regression according to equation

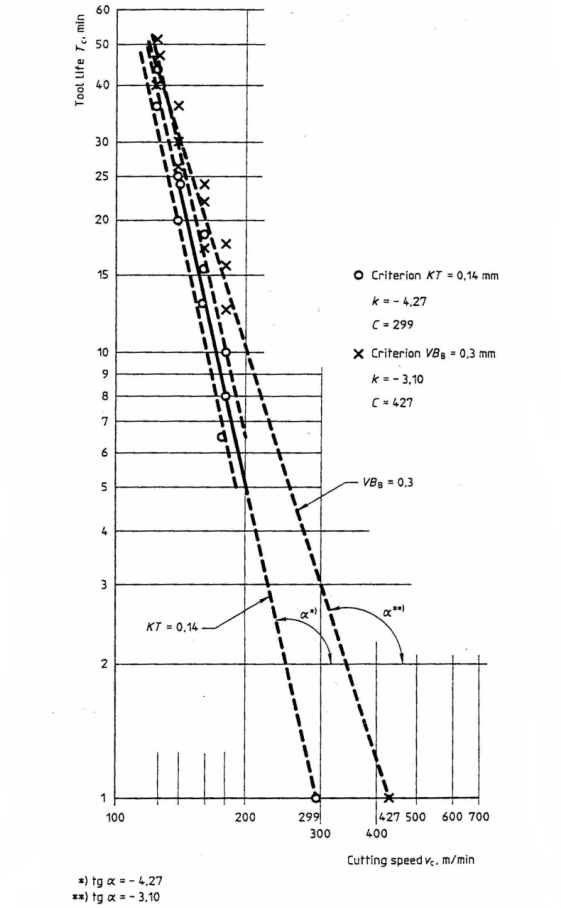


Figure 5: Tool-life curve fitting on logarithmic scale [1]

$$
\begin{aligned}
v_c \cdot T_c^{1/k} &= C, && (1)\\
v_c &\quad \text{cutting speed,}\\
1/k &\quad \text{Taylor's exponent,}\\
T_c &\quad \text{tool lifetime,}\\
C &\quad \text{constant.}
\end{aligned}
$$

Equation (1) is also called Taylor tool-life equation [1] or Taylor's basic equation [2, pp. 146-147]. It describes the linear section of the tool-life curve. However, Taylor's basic equation does not include the effect of cutting feed and is limited to certain range of speeds. Also it is not possible to know the wear of the tool at any time t during turning. Taylor's extended equation [2, p. 147],

$$t = \frac{C}{v^{1/n} f^{1/n_1} a_e^{1/n_2}}, \quad (2)$$

f_c cutting feed,

a_e depth of cut,

n_1, n_2 parameters.

include both feed and depth of cut, but it can only tell the tool life, not the wear. Also the parameters of the equation have to be estimated using data from experiments. Some of these drawbacks have been addressed in more advanced methods which will be discussed next.

Two different approaches are common in prediction of the tool wear. They are either based on a physical model [15] or the model is constructed from experimental data using, for example, neural networks [12]. Either way, some experiments have to be done in both approaches as physical models require experiments to determine parameters that affect the model. This is result the of being able to change the materials, *e.g.* the tool and work-piece, which have an effect in the model. Also, most physical models used in practice are linear, which means that they cannot properly model the whole tool life [15], [16].

Prasad *et al.* [12] use n artificial neural network (ANN) to predict both flank and crater wear. Cutting conditions as well as cutting time were given as inputs to the ANN. The outputs were flank and crater wear. Using 14 training patterns, the ANN was successfully trained and Prasad *et al.* come to conclusion that ANN is successful in predicting the wear. Error is less than 5 %.

Özel *et al.* [17] compare regressive modelling, *i.e.* curve fitting to data, and artificial neural networks in predicting surface roughness of the work-piece and flank wear of the tool. For the ANN, inputs were work-piece hardness, cutting speed and feed, length of the cut and cutting forces. Outputs were flank wear and surface roughness. Linear and exponential regression models were used while the ANN uses backpropagation to teach the neural network. Özel *et al.* note that ANN is better in predicting the surface roughness and wear than regression models and that the ANN is able to accurately predict the tool wear, with less than 5 % error.

Choudhury *et al.* [18] explore not only the prediction of surface roughness and flank wear

but also the temperature of cutting zone. Methods that were used were design of experiments (DOE) and artificial neural networks. DOE includes regression modeling. Cutting conditions were used as the inputs to the ANN, but time was not included. Choudhury *et al.* conclude that ANN is better in predicting or estimating the three output values than regression modeling with 5.66 % average error.

Danai *et al.* [19] developed a non-linear physical model that can represent the commonly known shape of the wear curve (shown in Figure 2). The model represents the wear as a function of cutting conditions and previous wear and it is differential model, that is, it models the speed of the wear. Total wear is calculated from the sum of abrasive and diffusive wear. Part of the model, the diffusive wear, is verified in [6] and [6] by Park *et al.*, with good results, the error is kept under 16 % of the measured wear. However, model was first linearized. The model can also accommodate changing cutting conditions during the machining.

Oraby *et al.* [20] study a mathematical model based tool life modeling compared to the extended Taylor's equation.. Especially the use of ratios between the cutting forces are explored with the use of the model. The model for wear incorporates cutting conditions and time as well as the cutting forces, in the form of their ratios. Also the tool life is modelled with same input variables. Average error in modeling is 12 % when tool life is over 13 minutes.

Choudhury *et al.* [15] developed a mathematical model (regression model) to predict the flank wear on a cutting tool. The wear is predicted from the cutting conditions. Parameters for the model were estimated from experimental data. Choudhury *et al.* conclude that the predicted results correlate well with measured values of flank wear.

Bhattacharyya *et al.* [21] use support vector regression to model tool wear in face milling. Bhattacharyya *et al.* use cutting force signals to predict the tool wear. The signals were collected during milling with corresponding wear measures. Support vector regression is successful in modelling the tool wear progression. Average error between predicted and measured values is lower than 5 %.

Shi *et al.* [22] model the flank wear on broaching tools using a support vector machine (SVM). Principal component analysis is used to determine which signal is suited best for predicting flank wear. The chosen signal was the cutting force. Using force, Shi *et al.* were able to predict the flank wear using SVM. Error in prediction was less than 0.01 mm in all test cases.

As can be seen from the review of existing methods for predicting the wear and tool life, many methods are based on either ANNs, regression analysis or support vector machines. Only few physical models, such as in [19], has been used to predict the wear. To be able to predict with an experimental data based method, a wide variety of experiments must be conducted for each tool - work-piece combination with different cutting condition combinations and still, the methods can only predict the tool life or wear only when the used cutting condition lie somewhere between the cutting condition combinations that were used to gather data for the method. This occurs as the methods can not extrapolate outside their data. In this sense physical models are better as they model the physical wear phenomena, and while they also require some experimental data to solve the model parameters, they can extrapolate better.

3 REQUIREMENTS OF THE VISUAL MEASUREMENT SYSTEM

This section describes the system design and argues for the choices made for the design. The topic of this thesis requires both hardware and software components, so the design section contains both aspects.

Initial requirements for the system were:

1. Measure the wear of the tool presented to the system.
2. Send the results forward to a system that uses the information at a higher level.

These are the two basic requirements that controlled the development of the system. One of the key issues is also that the system needs to operate during the turning process. This requires that the final system is robust enough to withstand the events that can occur during the process.

The original specification for both hardware and software can be found in Appendix 1. This specification was constructed from the initial requirements. Especially the turning center sets constraints for the whole system, as it was the system's working environment and the system had to accommodate it. Some changes had to be made, especially to the hardware specification and the changes to the original specification will be described in the implementation Section 6.

3.1 Hardware

Hardware design called for four distinct components: A camera, a lens for the camera, lighting and a computer, where the images are processed. These are also the basic components of a machine vision system. The components are shown here for clarity from the original specification in Figure 6.

Figure 6 presents 4 components and 3 communication channels between them. The camera component consists of camera, lens and lighting, these components handle the imaging. The vision computer processes the image data and measures the flank wear. The

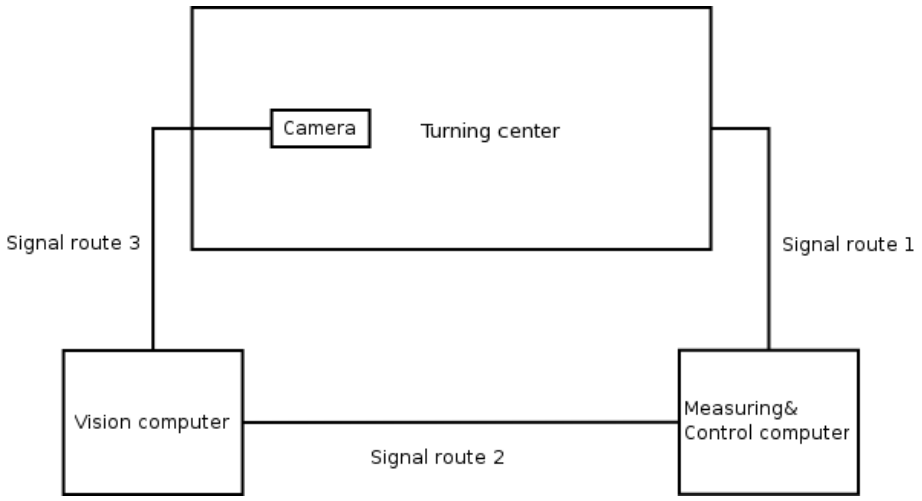
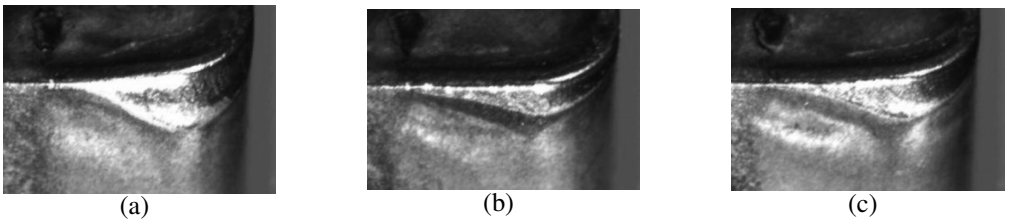


Figure 6: Hardware components

measurement and control computer is an external system that controls the visual measurement process and handles information about the turning process. The turning center executes the turning. Signal route 1 relays information from the turning center to the measurement and control computer about the position of the tool when it is in place for imaging. Signal route 2 relays the imaging command to vision computer. Route 2 also relays the measurement results from vision computer. Signal route 3 relays imaging commands to the camera and the camera relays image data to vision computer.

As seen in Section 2.2.1, specular reflection from the worn area was used in several cases to separate the unworn area from the worn area. This approach was selected also here. The validity of this method of lighting was first tested in preliminary experiments. Experiment was conducted with static lighting and static placement of the tool. Also the camera position was static over all images. Sample images from the experiment are shown in Figure 7.



(a) (b) (c)

Figure 7: Tool wear at different stages, illuminated by a spotlight

Images in Figure 7 show that there is some specular reflection from the worn area at different levels of wear, these are visible as bright spots on the images. This meant that the approach of using reflected light as a method of detecting the worn area was viable

and, thus, was included in the specification found in Appendix 1.

Camera specifications were based on the analysis of the previous applications, which are described in 2.2.1. In previous applications the range of resolutions, with CCD sensor cameras, were between $3\ \mu m$ and $12\ \mu m$ per pixel. This is the physical resolution, *i.e*, the resolution at which the camera sees the target object. Considering the specified imaging area ($10 \times 7\ mm$), the physical resolution was set to at most to $10\ \mu m \times 10\ \mu m$ per pixel. Requirements dictated that lens must have a working distance of $20\ cm$ to $30\ cm$, *i.e.* focusing range. This was measured during initial checks of the turning center [23] and it was a safety requirement so that the flying chips and continuous chips from the work-piece would come into contact with the camera and lens as little as possible.

Connections from the vision computer to remote components such as the camera, were chosen in part of convenience and ease of use. This is especially true with the selection of TCP/IP and Ethernet protocols which implement the communication with remote computers. More on TCP/IP and Ethernet in Section 6.

Firewire bus was chosen for the camera connection, because it bypasses the need for traditional camera grabber as the image data is sent digitally and available Firewire interfaces could be used in the software of the system. Firewire bus also carries power. So choosing Firewire, the number of additional components could be minimized to essentials, eliminating the need for additional hardware in addition to the computer and camera. Firewire bus also allows changing of the camera settings, such as shutter time, directly from software.

Specifications for the vision computer itself were chosen so that the computer would be faster in processing the images than the development computer. This ensures that when the development computer processes the image faster than required, also the vision computer will do so.

3.2 Software

The system relies heavily on software. The software will be located in the vision computer. The requirements described before, require that the software must be able to measure the wear from the images captured by external cameras. Also the software needs to send the results of the measurement to remote location. There are two main areas

which the software had to implement: image processing and communication with remote computers and users.

Image processing handles the measurement of the tool. The measurement was focused on the measurement of the flank wear. Crater wear was specified secondary. Flank and crater wear measurements were specified to conform with standard [1]. The measurements could then be compared to manual wear measurements.

Also time constraints were placed on the software processing of the image. These constraints were derived from the cutting intervals of the turning center. These constraints were placed to allow stopping of the turning center before a new cut from the work-piece was started.

Communication was required to be bi-directional between the vision computer and measurement and control computer. This allows the measurement to be controlled, *i.e.* triggered, when the tool is moved to be imaged and the results of the measurement could be sent to the remote computer for the higher level controller. It was required that communication with the user of the software may only occur during the initialization of the software to allow maximum automation. User communication is necessary to relay parameters for tool measurement.

4 MEASURING TOOL WEAR WITH MACHINE VISION

In this section the approach which was developed to measure flank wear is described. Implementation details are discussed in Section 6. The theory behind the methods used in the algorithm is explained to clarify the actual algorithm. A brief introduction to machine vision in general is also given.

4.1 Machine Vision

Machine vision is used to *"Recover useful information about a scene from its two-dimensional projections"*, as quoted from [24, p. 1]. This quote embodies the goal of any machine vision system. The quote also sets some constraints on the machine vision system. To get useful information, it must first be decided what is useful information and to get the information at all, vision must be used so that the information is available to be recovered from the projection, *i.e.* the image. In machine vision, information recovery is usually left to a dedicated processor or a computer, but the processor can be thought of as a human doing the same processing, *e.g.* measuring wear from the tool, like in the context of this thesis. Compared to a human, machine based vision processing has many benefits, like repeatability, which has led to many applications used in industry and military.

The components of a basic machine vision system can be seen in Figure 8. The Basic components of any machine vision system are: Camera, frame grabber, lighting, object and processor. These basic components allow the system to image the object, *i.e.* capture 2D representation of the object, and extract the useful information from the image.

The camera is combined with a lens. Using a lens, the camera can be focused to different distances. Each lens has a specific *focal length* or a range of focal lengths, if the lens is a zoom lens. Focal length determines object's size in the 2D projection. The camera and lens are usually modelled with a *perspective projection* [25, p. 4], which will be used in this thesis as well to model the camera and lens.

The Perspective projection is a model that applies when a pinhole camera is used. The pinhole is presumed to be so small that only one light ray for each point in the scene is able to pass the pinhole. This is not true in the real world, but perspective projection is a

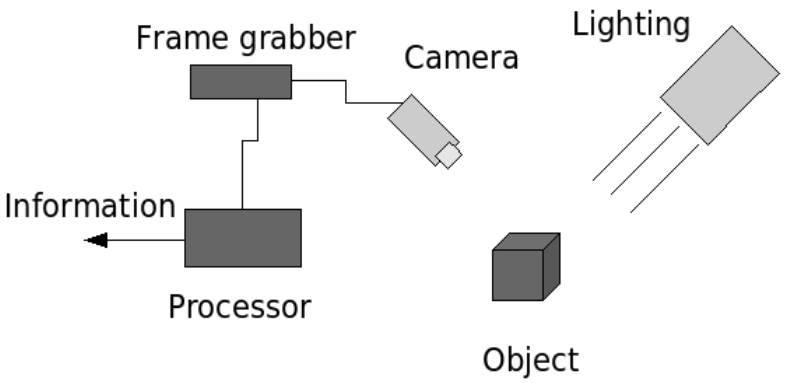


Figure 8: General machine vision system

convenient model for both real pinhole cameras and cameras with lenses. [25, p. 4]

A frame grabber is used with a camera to convert an analog signal from the camera to digital images. This process involves analog-to-digital converters which quantize the analog signal received from the camera. In modern cameras, the process is usually integrated to the camera, which can output the digital image through a digital bus.

The lighting set-up depends entirely on application. There are multitude of different light sources. Two often used light sources are point light sources and uniform light sources. Point light sources can be used to create specular reflections, as in many of the measurement applications discussed in Section 2.2. Uniform light sources create uniform lighting, removing any specular reflections, which can be useful in many applications, *e.g.* detecting texture of a surface.

The object represents the observed object. Other components of the system, lighting and camera, have to be chosen such that the information can be recovered. For example, if the object is large, a wide-angle lens and powerful lighting might be needed to image the whole object with sufficient lighting.

Finally, the processor takes the image and performs image processing to recover the required information from the image. The processor is usually a computer attached to the camera, but the image processing can be done in the camera as well.

4.2 Algorithm for Measuring Flank Wear

As said in Section 3, one of the most important requirements for the system was to measure the flank wear. This section describes the algorithm used to detect and measure the flank wear from tool images in detail. Focus is on the theoretical aspect of the algorithm and implementation details are covered in Section 6. Methods used in the algorithm itself are also described in the subsections for clarity.

4.2.1 Algorithm

The algorithm presented here requires certain conditions to be met. The most important of these is the presence of a specular reflection from the worn area of the tool, as discussed in Section 3. Second one is that the whole major flank of the tool is visible in the image. The flank image is required for calibration purposes. An example of an acceptable tool image is shown in Figure 9. The edges are referred to horizontal edges and vertical edges in the algorithms that follow. In practice it is possible to tune the algorithm so that edges in any angle can be considered to be horizontal or vertical, but for the purpose of demonstration, the example image is the base assumption.

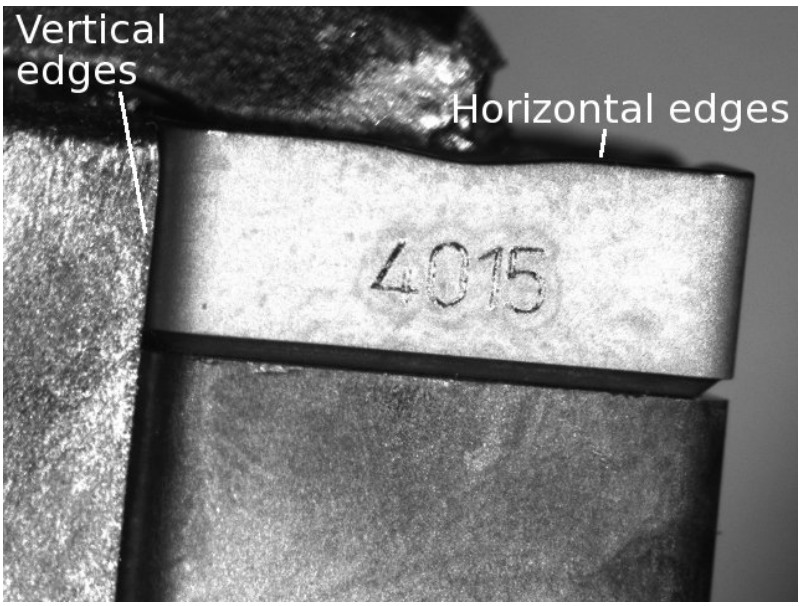


Figure 9: An example tool image

The high level algorithm for measuring the wear can be seen in Figure 10. It describes the high level functions of the algorithm which includes both calibration and measurement.

1: **Algorithm measure_tool():**
2: $firstimage$ = TRUE
3: **for** each command to image **do**
4: img = new tool image
5: **if** $firstimage$ = TRUE **then**
6: $noseregion, pxmm, pymm$ = **calibrate(**img**)**
7: $firstimage$ = FALSE
8: **end if**
9: $\text{VB}_C, \text{VB}_B, \text{VB}_B max$ = **measure_wear(**$img, noseregion, pxmm, pymm$**)**
10: return $\text{VB}_C, \text{VB}_B, \text{VB}_B max$
11: **end for**

Figure 10: High level algorithm for measuring wear

The algorithm in Figure 10 has two main parts, a calibration function and a wear measurement function. The calibration is run only at the start, this determined with the variable $firstimage$ at line 5. If the image is the first one that was imaged then **calibrate()** function is run. New image that was taken at line 4, is passed as an argument for the calibration function. Calibration returns the result of calibration, in this case how many millimeters is one pixel in the image, in both x- and y-axis. The calibration also returns the region coordinates, $noseregion$, that indicate where the actual wear is for further processing and measurement. This step reduces the needed computation considerably, as only the interesting part of the tool has to be processed. The algorithm for calibration follows in Figure 11.

The goal of the algorithm in Figure 11 is to determine the calibration of the system and support the measurement of the flank wear by also determining the area around the flank wear. The algorithm begins by applying basic image processing operations in lines 2-5. Image is converted to grayscale to ease further processing. After conversion, a series of sharpening and smoothening operations is done. This guarantees that the actual tool edges have a high image gradient. The smoothing operation smooths rough edges that may have formed during the brightness and contrast adjusting. Sharpening ensures good results from the edge detector, which is used in line 6.

Goal of the next section in the algorithm is to find the tool edges. Using the edge pixels from Canny edge detector, the pixels are transformed in to Hough transform space using the Hough transformation in line 7. After this, a selection of the most strongest lines in the Hough space are chosen. Selection of the lines is made so that lines are chosen from both vertical and horizontal angles, as the tool is rectangular in the image and the tool edges should be quite near to either horizontal or vertical angle. Using The Hough transform,

1: Algorithm **calibrate**(img)**:**
2: Convert img to grayscale image $gimg$
3: Adjust $gimg$ brightness and contrast to sharpen the edges of the tool
4: Apply Gaussian filter to $gimg$ to smooth sharp edges
5: Adjust $gimg$ brightness and contrast to sharpen edges
6: Convert $gimg$ to edge image $eimg$ by using Canny edge detector
7: Convert $eimg$ to Hough transform space image $himg$ using Hough transform
8: Find most prominent lines from $himg$, from both near horizontal and near vertical angles
9: Find 4 lines from $himg$, by selecting 2 furthest lines from each other from the horizontal and vertical angles
10: Using the 4 lines, calculate their intersections, representing tool shape
11: Using the intersections and known tool dimensions, calculate mm per pixel ratios, $pxmm$ and $pymm$
12: Using the intersections, determine the intersection representing the intersection of major and minor flank at the top of the tool
13: Calculate region $noseregion$ around the intersection for measuring the flank wear, determined in line 11, using known cutting parameters
14: return $noseregion$, $pxmm$, $pymm$

Figure 11: High level algorithm for calibration

the angle of the tool can change and the edges of the tool can still be found. More on the Hough transform properties will be said in Section 4.2.4.

After the edges of the tool have been found, intersections of these lines are calculated. Figure 12 shows the intersections determined from the tool edge lines. Determining the intersections is a simple operation, the intersections can be easily solved, once the lines are in slope-intercept form,

$$y = ax + b, \tag{3}$$

however the line parameters in Hough space differ somewhat from the analytical form parameters a and b. In this case Hough transform tells the distance of the line from the image center, ρ, and the angle of the line θ. In this particular case, the θ is perpendicular to the actual line, the parameters a and b are solved in essence as follows,

$$a = \tan(\theta - \frac{\pi}{2}), \tag{4}$$

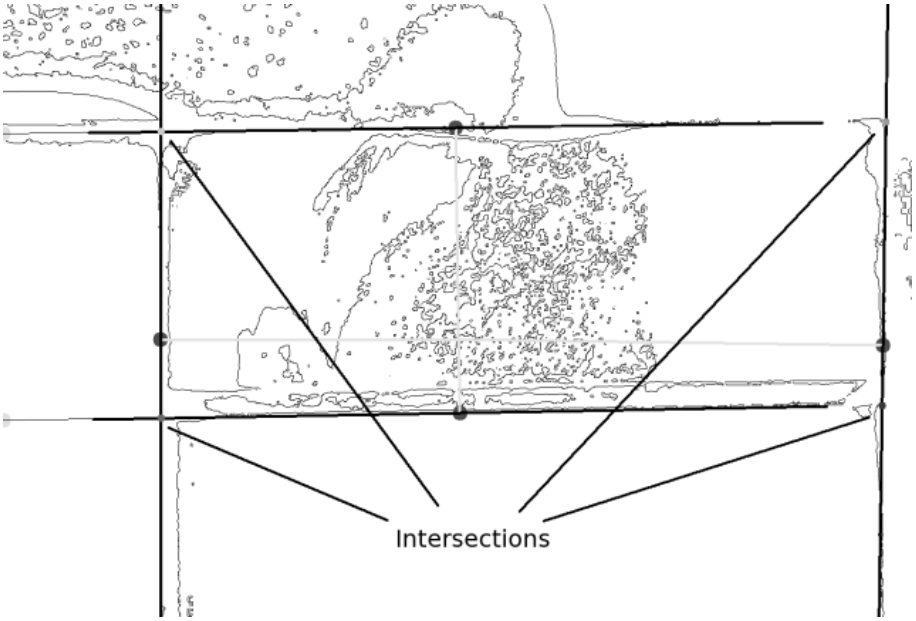


Figure 12: Tool edges intersecting in edge image of the tool

a can be easily solved as it determines the steepness of the line slope, *i.e.* the angle. This can be solved using the tangent function $\tan$ on the angle of the line. Parameter b describes the intersection with the y-axis when $x = 0$,

$$b = M_x \cdot \tan(\frac{\pi - 2 \cdot \theta}{2}), \tag{5}$$

where,

$$\begin{aligned} M_x &= C_x + \rho \cdot \cos(\theta), \\ M_y &= C_y + \rho \cdot \sin(\theta), \end{aligned}$$

here C_x and C_y denote the image center coordinate. Calculation of b is based on right triangles which can be seen in Figure 13. By knowing angle θ, the angle α can be solved. Also the point (M_x, M_y) is known.

After the parameters of all lines have been solved, solving the intersections between the lines is trivial,

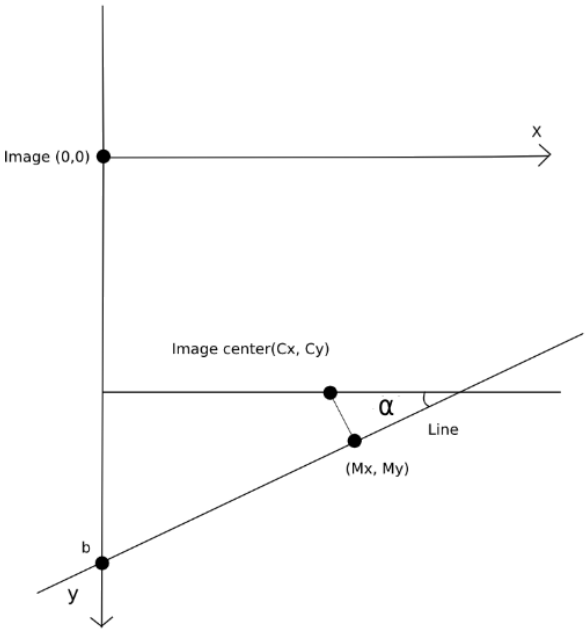


Figure 13: Calculation of b

$$\begin{aligned} a_1x + b_1 &= a_2x + b_2, \\ x &= \frac{b_2 - b_1}{a_1 - a_2}. \end{aligned} \tag{6}$$

Using equation (6), the x-coordinate of the intersection can be solved and then the y-coordinate can be solved using equation 3. After all the intersections have been solved, the four points represent the shape of the tool, which should be rectangular. After the calculation of the sides of the rectangular shape, representing the length and width of the tool, the calibration of the camera can be determined. In the algorithm in Figure 11, this is done in line 11. Calibration can be done by finding the ratio between the image distance, *i.e.* image pixels, and the actual metric measures of the tool. Result of this is a ratio between millimeters and image pixels, $\frac{px}{mm}$. These are saved as $pxmm$ and $pymm$ to account for non-square pixels in the camera.

After the calibration is determined, the calibration algorithm finds the intersection representing the intersection of the major and minor flank where the actual wear begins to form. This is entirely a practical issue, depending on the implementation of the algorithm in Figure 11, and as such, it is discussed in Section 6. After the intersection has

been determined, the algorithm determines the tool nose region, $noseregion$. This image region can be determined with the help of known parameters, especially cutting depth b and cutting edge angle, K_r. As the calibration is known the cutting depth can be directly calculated as pixels in the image. Using these known parameters ensures that the wear is visible inside the image region. There are a number of considerations regarding the projection of 3D space to 2D image plane. Some of these are discussed in Section 4.2.5.

After $noseregion$, $pxmm$ and $pymm$ have been calculated, they are returned and calibration is complete. As the algorithm for whole measurement process in Figure 10 states, the calibration is run only once. After calibration, only the main measurement algorithm is run, as shown in the algorithm in Figure 10. This can be done under the assumption that the tool remains at a static pose between the images, so that calibration results apply in each image. The algorithm for measuring the wear is found in Appendix 3. Image processing is done in similar manner to the calibration, but goal in the measurement algorithm is to find the tool edge above the flank wear, after which the wear can be measured with the calibration data provided by the calibration algorithm.

Measurement algorithm, in Appendix 3, begins with exactly same operations as the calibration algorithm. The image, img, brightness and contrast must be adjusted so that the edges can be detected, although this step can be passed if edges are clear in the original image. However, the edges are shorter in the img, as the algorithm only handles the image region, received from the calibration, there is no need for filtering the image with a smoothing filter, because the tool edges do not have to stand out from noise as they are the only strong edges in the image region.

After the $eimg$ has been formed using the Canny edge detector, it is transformed to a Hough transform space. This is the same procedure as in the calibration algorithm. Difference is that only two lines are searched for in the Hough space, as only one horizontal edge and one vertical edge of the tool are needed to determine the tool location in the image region. Again, the process of determining the intersection of the two lines is exactly the same as in the calibration algorithm. These steps, lines 2-8, represent the geometric operations used to find the location of the tool in the image.

After the actual location of the tool in the image has been found out, the problem changes to determining the worn area from the tool image. This is achieved by first extracting the blue channel from the RGB image. Blue channel was chosen as the light used in the system has a blue tint. Of course it possible to use a channel of choice or a gray-scale image of the tool for the same purpose. After the individual channel is extracted, Gaussian

smoothing is applied to smooth the illuminated worn area, so that there is less possibility to detect false edges in the worn area.

After smoothing the image, the width of the cut is calculated directly as pixels in the image along the horizontal edge. Using the tip of the tool, given by the intersection of the two edges of the tool, and the point at the distance of the known cutting depth from the tip, the pixels between them in a straight line can be solved. Common algorithm for this is the Bresenham's line algorithm, this is described in detail in Section 4.2.2. Bresenham's line algorithm is applied line 12 of the wear measurement algorithm. The obtained line is used as a base for determining the worn area as it defines the upper boundary for the wear. A small change for the Bresenham's line algorithm was required so that the order of the pixels on the horizontal edge remains the same, even if the angle changes. The modification was needed as the Bresenham's line algorithm relies on symmetricity and mirroring, which can flip the direction of the pixels.

From this upper boundary, a series of lines in the same angle as the vertical tool edge are cast to a defined lower boundary. The lines are cast from each pixel of the upper boundary. All these lines are inspected for the worn area edge. This inspection is done by selecting an array of values from the single line from the image $bimg$ according to the median mask size and a median value is calculated for each pixel in the line. The edge of the worn area along this single line is detected by comparing the median values from the previous pixel to current. If this median value changes more than a predefined threshold, the edge of the worn area is marked. After all lines from the upper boundary have been inspected, the whole worn area edge to the depth of cut is available for further computation.

Using the worn area edge, the values VB_C, VB_B and VB_B max. can be solved. This process corresponds directly with the measurements in the ISO standard [1]. VB_C is the largest value, *i.e.* the largest wear value in the region C. VB_B is the average value of the wear in region B and VB_B max. is the largest value in region B. Using the conversion constants received from the calibration algorithm, the values of each of the measures can be directly changed to the metric system, *i.e.* to millimeters. After the measures have been returned, the algorithm is finished.

4.2.2 Bresenham's line algorithm

Bresenham presented a complete line algorithm in [26]. Bresenham's algorithm is an efficient algorithm to draw lines as the algorithm operates without multiplication or division. The algorithm is also well known as it was first published in 1965. [26]

The algorithm, which is used in the measurement algorithm, is a version of the Bresenham's line algorithm, which uses only integers to calculate the locations of the pixels in the line. The algorithm is presented in Appendix 3.

The actual algorithm is quite simple. Main body of the algorithm is found on lines 20-31. The algorithm is based on the error generated by quantizing the line into pixels. When this error grows enough to zero or above in the algorithm, a step is taken in y-direction, meaning that the pixels are one step above or below the last plotted pixel. This, in essence, is the algorithm which determines the y-coordinate corresponding to a certain x-coordinate in a discrete grid as the x-coordinate can be found simply by incrementing it after each plotted pixel. Example of the algorithm is found in Figure 14. Gray squares depict the plotted pixels in the grid along the true line.

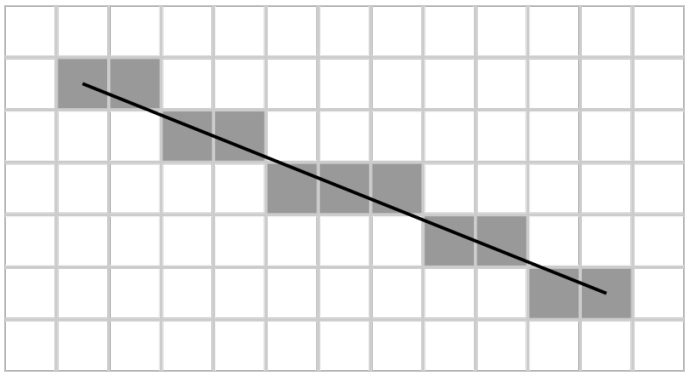


Figure 14: Example of Bresenham's line algorithm

The first part of the Bresenham's line algorithm, lines 2-19, is mainly initializations to force the line drawing problem to the first octant of the x-y-coordinate system, *i.e.* the slope of the line is between 1 and 0. The line is forced to the first octant by swapping the given coordinates, if the line slope is steep or the first x-coordinate is larger than the second. Also, $ystep$ is properly chosen, so that lines with negative slope can be drawn.

4.2.3 Canny edge detector

Canny edge detector was first published in 1986 by Canny [27]. The Canny edge detector gives near optimal results on edge detection. The algorithm for Canny edge detector involves multiple steps to ensure a good localization of the edges. The algorithm is presented in Figure 15. [24, pp. 169-172]

1. Smooth the image with a Gaussian filter
2. Compute the gradient magnitude and orientation using finite-difference approximations for the partial derivatives
3. Apply nonmaxima suppression to the gradient magnitude
4. Use the double thresholding algorithm to detect and link edges

Figure 15: Canny edge detection [24, p. 172]

Canny edge detection algorithm is divided into 4 clearly separated operations. Gaussian filtering is performed by convolving the filter over the image,

$$S[i, j] = G[i, j; \sigma] * I[i, j], \tag{7}$$

where G is the Gaussian mask and I is the original image. Gaussian mask can be 1 or 2 dimensional, with 1-dimensional array, the Gaussian mask can be convolved horizontally and vertically over a image as the convolution is separable [24, pp. 129-130].

Values for the 2D Gaussian mask G can be calculated directly from the Gaussian distribution,

$$G[i, j] = ce^{-\frac{(i^2+j^2)}{2\sigma^2}}, \tag{8}$$

where c is a normalizing constant. Using equation (8), values can be calculated directly for each array cell. The array can then be used in the convolution. [24, p. 132]

The second step of calculating image gradient magnitudes and orientations, is used for determining the direction and strengths of the edges in the original image I. The magnitude and orientations are calculated with the help of partial derivatives, which are obtained by a finite-difference approximation from S,

$$P[i,j] \approx (S[i,j+1] - S[i,j] + S[i+1,j+1] - S[i+1,j])/2 \quad (9)$$
$$Q[i,j] \approx (S[i,j] - S[i+1,j] + S[i,j+1] - S[i+1,j+1])/2 \quad (10)$$

where P and Q are approximations of the partial derivatives. Using P and Q, the magnitude M and orientation θ for each pixel is calculated,

$$M[i,j] = \sqrt{P[i,j]^2 + Q[i,j]^2} \quad (11)$$
$$\theta[i,j] = arctan(Q[i,j], P[i,j]), \quad (12)$$

after the magnitude and orientation have been calculated, nonmaxima suppression is performed. [24, pp. 168-170]

Goal of nonmaxima suppression is to find the edges from the magnitude array by suppressing all other than the local maxima in the array, *i.e.* zeroing all other values in the array. End result of nonmaxima suppression is the actual edge information, array N. This is achieved by first quantizing the orientation to four sectors. After quantization, each magnitude is compared with its neighbours along the sector. If the magnitude is the local maximum, it is retained, otherwise it is removed. [24, pp. 170-171]

Thresholding is performed on the array N. Canny edge detector uses double thresholding to find true edges. This produces two images, T_1 and T_2. T_2 can contain gaps in the true edges, while T_1 might have noise, *i.e.* false edges. Idea behind double thresholding in Canny edge detector is to combine T_1 and T_2, so that gaps in T_2 are filled from T_1, by checking the 8-neighbourhood of edge in T_2 from T_1 and filling the gap until it is completely filled. [24, p. 172]

4.2.4 Hough transform

Hough transform is a global image processing method, *i.e.* Hough transform acts on the whole image at the same time. Methods like Canny edge detector are local image processing methods acting only on small image patches. Main use for Hough transform

is to detect lines from images, although it can be generalized to any shape, like circles or ellipses. Hough transform was published in 1962 by Hough. Hough transform is used in the algorithm described in 4.2 as it can detect global lines, *e.g.* tool boundaries, from the image, even when there is a large amount of noise in the image. [28, p. 587] [29, p. 362]

1. Quantize the parameter space appropriately.
2. Assume that each cell in the parameter space is an accumulator. Initialize all cells to zero.
3. For each point (x, y) in the image space, increment by 1 each of the accumulators that satisfy the equation.
4. Maxima in the accumulator array corresponds to the parameters of model instances

Figure 16: The Hough transform algorithm[24, p. 220]

As can be seen from the Hough transform algorithm in Figure 16, the concept behind Hough transform is to change the problem of detecting lines in the image to detecting maxima in parameter space. In practice, a model has to be known, in this case a model for line. Parameters of this model form the parameter space. In the original publication, model for the line was the traditional slope-intercept form, shown in equation 3. The parameter space in this case is (a, b) instead of the image space of (x, y). However, the slope-intercept form has problems when lines have near vertical angles. This results in near-infinite a. [28, pp. 587-588]

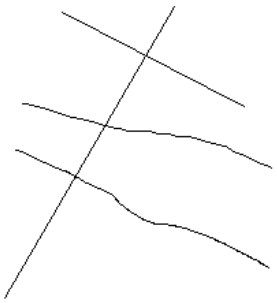
The problem can be bypassed by using a different model for the line. A suitable model is the polar representation of the line,

$$\rho = x \cos\theta + y \sin\theta. \tag{13}$$

This model can represent all lines, but in polar coordinates. The parameter space is (ρ, θ). [28, p. 588]

Hough transform requires that the parameter space is discretized. This reduces computation as only finite amount of parameter combinations are possible. Each discrete position in the parameter space is an accumulator. Because of this structure, Hough transform is also called a voting method, which reflects the inner working of the Hough transform. Each pixel votes for each individual cell in the parameter space which satisfies the equa-

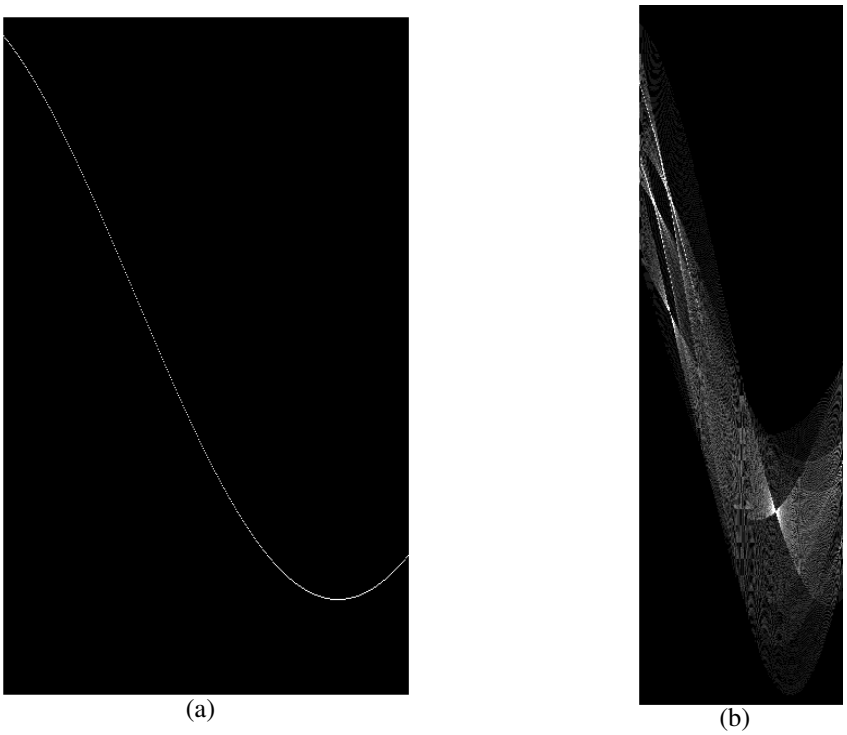
tion 13, with known (x, y). Example images can be seen in Figure 17.



(a) (b)

Figure 17: (a) Image with only one active pixel; (b) Image with multiple lines.

Images in Figure 17 are shown in the parameter space (ρ, θ) in Figure 18. The figure shows the accumulation of multiple pixels as bright spots *i.e.* maxima. These bright spots indicate a line in the image original image with parameters (ρ, θ) of the spots.



(a) (b)

Figure 18: (a) Fig. 17a) after Hough transform; (b) Fig. 17b) after Hough transform.

4.2.5 Considerations

A few considerations must be acknowledged when working with a visual measurement system. When the considerations are known, the system can operate more reliably and accurately directly affecting the measurements.

One consideration that clearly affects the measuring relates to the object. If the object to be imaged is not parallel to the imaging plane, the projection does not correspond fully to the actual measures of the object. The problem is visualized in Figure 19.

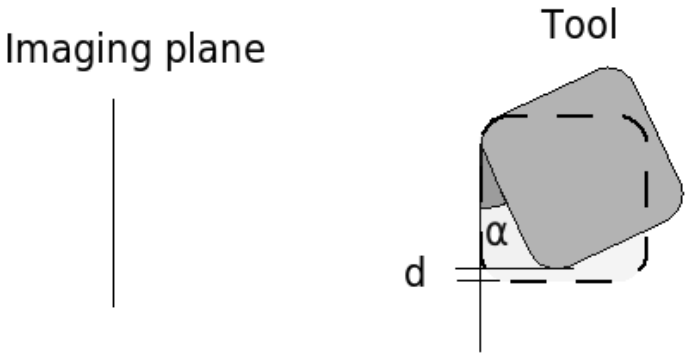


Figure 19: Image plane and object.

However if the angle, α between the object and parallel plane is known, the difference d between the perceived object length and actual object length in projected image can be calculated. The calculation can be done if the object dimensions are also known. One way of determining d,

$$\begin{aligned} d &= l_2 - l_2 \sin(\alpha - \frac{\pi}{2}), \\ l_2 &\quad \text{Length of the tool} \end{aligned} \tag{14}$$

gives the difference of metric length from the perspective of the imaging plane between fully parallel and skewed objects. With this information it possible to calculate the pixel-per-mm ratio using the object's actual length in the image instead of the known measure of the object's length.

5 MODEL BASED PREDICTION OF TOOL WEAR

This section describes the background for the modeling of tool flank wear and goals for its prediction. Three methods are introduced, which can model the flank wear progression. The methods are linear regression, artificial neural networks and support vector regression. These methods are also tested in Section 7.2.

5.1 Modeling Flank Wear

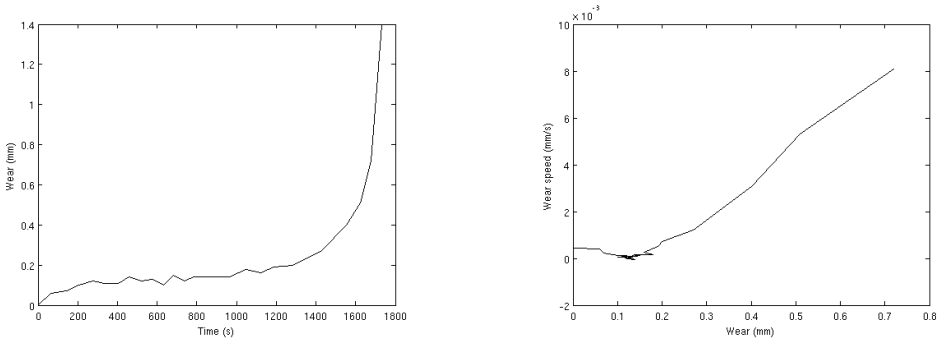
There are multiple ways to close-in on the problem of modeling the tool wear. Previously explored methods are discussed in Section 2.3. The most commonly used approach, *e.g.* [15], [12], is to model the wear in relation to cutting time or cutting length, cutting conditions (V, f, b) and possibly other measurements, such as cutting force. This results in a model that can predict the wear at given cutting time t or cutting length d. With these models, it is not possible to model the wear under changing cutting conditions.

The only way to solve the problem of changing cutting conditions is to use a differential approach to model the tool wear progression. With a differential model, the tool wear is not modelled as such, but in its speed. To calculate the speed, the critical information is the current tool wear in addition to the cutting conditions and other possible sensory data. Reason for this can be seen in Figure 2. The speed of the wear, *i.e.* the wear differential, is dependent on the amount of wear. So, to be able to model the wear differential, current tool wear information is needed.

Only a few models using the differential approach have been developed [16], [19] compared to the models which do not use differential modeling. In [6], Park *et al.* verify that the differential model they use is able to handle changing cutting conditions, in this case cutting feed. In this light, using a differential model in prediction of the flank wear is at least one of the methods that can handle cutting condition changes. Even though the model in [6] is a physical model, the idea of using a differential model can be applied to methods represented in sections 5.3-5.5, which are based solely on the data about the turning process and wear.

The problem that arises from using, for instance, ANNs, is that no data can be collected of the wear differential, *i.e.* wear speed. However, wear differential can be estimated from known data, cutting time and flank wear level at those cutting times. The most

simple method is to fit a line to the time-wear coordinates for each data point. This can be achieved, for example, with linear regression. An example of a wear curve and its corresponding wear speed, is shown in Figure 20.



(a) (b)

Figure 20: (a) Wear curve; (b) Wear speed.

Instead of using the wear width data directly as a variable, wear speed is used in the model. This accomplishes the goal of modeling the wear differential as mm/s.

5.2 Flank Wear Prediction With A Differential Model

With the selection of the differential model, a few complications rise. Selecting to model the wear speed instead of the wear, flank wear cannot be directly determined from the model anymore. To predict the flank wear at certain cutting time t requires an iterative approach to be used in conjunction with the wear model. An algorithm to calculate the wear at given time t, is given in Figure 21. The algorithm is simplified in this case, as no information of cutting conditions is needed.

1: **Algorithm predict_wear(**t, $offset$**):**
2: $total\_wear = 0 + offset$
3: **for** $i = 1$ to t **do**
4: $diff\_wear =$ wear_model($total\_wear$)
5: $total\_wear$ = $total\_wear$ + $diff\_wear$
6: **end for**

Figure 21: Algorithm to determine wear at time t

The algorithm presented in Figure 21, calculates the total wear width, $total\_wear$, by using wear_model-algorithm in line 4, to determine the wear differential $diff\_wear$ from the current wear width. The wear_model-algorithm can be any algorithm that can return the wear speed from the current wear width or other information, such as cutting

conditions or other data. This includes physical models or models which are based, for instance, on ANNs. The time t, given in the first line of the algorithm as an argument, is presumed to be integer seconds.

With the algorithm in Figure 21, it is now possible to transfer the information about the wear speed to concrete wear width information at any time t. Next, a few methods that can form the core of the predict_wear, *i.e.* the wear model, are presented. Experiments and results obtained from the experiments using the presented methods are discussed in Section 7.2.

5.3 Linear Regression

Regression is a statistical method for analyzing data. The data consists of dependent and independent variables. The dependent variable depends on the independent variables. For example, in the context of this thesis, the independent variable is the wear width and the dependent variable is wear speed. Linear regression takes the data and models the relationship of the variables with a linear equation. With the knowledge of the parameters, it is possible to predict the dependent variable using previously unknown independent variable values. [30, pp. 211-212]

Simple linear regression, linear regression with only one independent and dependent variable, is described by

$$\begin{aligned} Y_i &= \beta_0 + \beta_1 x_i + \epsilon_i, \qquad (15) \\ Y_i, &\quad \text{dependent variable,} \\ x_i, &\quad \text{independent variable.} \end{aligned}$$

β_0 and β_1 are unknown constants that have to be estimated using available data. ϵ_i are random variables, which describe error in data as $N(0, \sigma)$. Generalized linear regression with any number of independent variables, can be achieved by adding a constant term for each variable into the equation. [30, pp. 211-213]

The method used commonly to estimate the constants β_0 and β_1 is called least squares

estimates. This methods works by minimizing the residuals, e_i, which are estimates of the errors, ϵ_i. Substituting β_0, β_1 and Y_i with b_0, b_1 and y_i, residuals can be solved and minimized,

$$e_i = y_i - (b_0 + b_1 x_i), \tag{16}$$

$$min \sum_{i=1}^{n} e_i^2. \tag{17}$$

From equation 17 it is possible to solve the parameters b_0 and b_1 which minimize e_i, *i.e.* provide the best possible fit to the data,

$$b_0 = \bar{y} - b_1 \bar{x}, \tag{18}$$

$$b_1 = \frac{\sum (x_i - \bar{x})(y_i - \bar{y}}{\sum (x_i - \bar{x})^2}. \tag{19}$$

After the parameters b_0 and b_1 are solved, it is possible to solve estimation of y_i, $\hat{y_i}$. [30, p. 215]

5.4 Artificial Neural Networks

Artificial neural networks are loosely based on the idea of human brain. The network consist of units, neurons or perceptrons, that are connected to each other with weighted inputs and outputs. An example of a multilayer neural network is shown in Figure 22. The number of hidden layers and neurons in the hidden layers can be decided freely, however, the numbers of input and output layer neurons are tied to the number inputs and outputs required from the network. Other types of neural networks exist, but this section concentrates on multilayer perceptrons. [31, pp. 284-285]

Each input neuron is connected to the input data. Each input neuron then feeds the inputs to the hidden layer. Hidden layer neurons then feed other hidden layers or outputs. This operation is called feedforward operation. Each hidden and output layer neuron weights its own inputs and sums the inputs. This forms a net activation value. This net activation

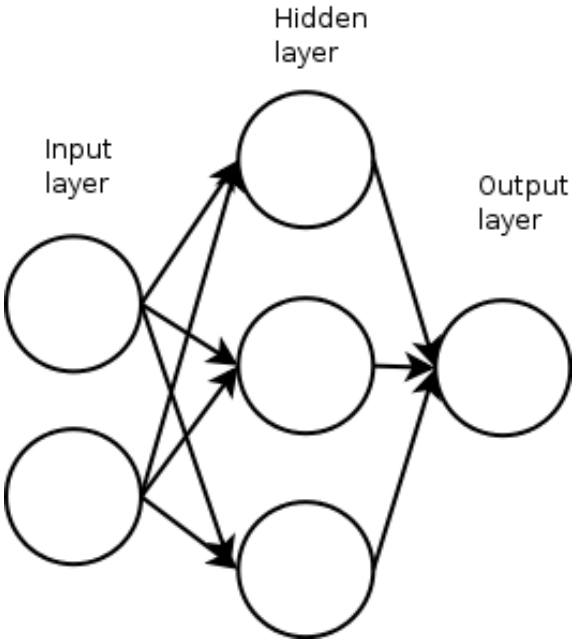


Figure 22: Example of multilayer networks.

value is then fed through a non-linear function, which forms the final output of the neuron. [31, p. 285]

From the previous description of the neural networks operation, it can be seen that the whole network is based on the individual weights of the neurons. These weights are learned through supervised learning. Supervised learning requires that corresponding outputs are known for each input which is given to the network. The weights are then tuned iteratively so that the error between known output and network output decreases. These iterations are called epochs. The training method is known as backpropagation. Gradient descent is the basic method used to decrease the error through backpropagation learning, which adjust the weights of the perceptrons from the output layer back to the input layer. [31, pp. 288-290]

Neural networks have many uses, although modeling and classification are the main ones. Neural networks can be adapted to large number of problems, but ANNs require training data. Negative aspects of neural networks are their inability to extrapolate, *i.e.* they cannot predict phenomena that are outside of their training data. Also there is no clear method of choosing the parameters of the network, *e.g.* number of layers or neurons. Neural networks are considered to be black boxes, *i.e.* the relation between input and output cannot be easily analyzed.

5.5 Support Vector Regression

Support vector regression is based on the support vector machines (SVM). The support vector machine was introduced by Cortes and Vapnik [32] in 1995. SVM is a maximum margin classifier, *i.e.* SVM finds the hyperplane that has maximum margin to different classes. To find the hyperplane, SVM uses a kernel that transforms the problem of classifying to a higher dimension. The kernel on SVMs can be changed, which gives the possibility of building a nonlinear classifier. The name "support vector machine" comes from the use of vectors, *i.e.* data points, that describe the maximum margin. [32]

Support vector regression (SVR) takes the ideas behind SVM, but implements them on a regression framework. SVR is an extension to the SVM. So instead of finding the hyperplane that divides the data into discrete classes, the SVR uses regression to find the best fit of independent variables to dependent variables, much like in linear regression. The support vectors now describe the margin as well but from a regression viewpoint. The margin is also called the ϵ-tube. [33]

6 VISUAL MEASUREMENT SYSTEM IMPLEMENTATION

This section describes the implementation of the machine vision measurement system as well as the implementation of the tool wear modelling. As the measurement system requires both hardware and software components, both aspects are explored here.

6.1 Hardware components

Hardware components are the key factor in the implementation of the measurement system. This section discusses the implementation of the hardware design issues presented in Section 3. Overview of the whole system is given in Figure 23 with communication channels between individual elements in the measurement system.

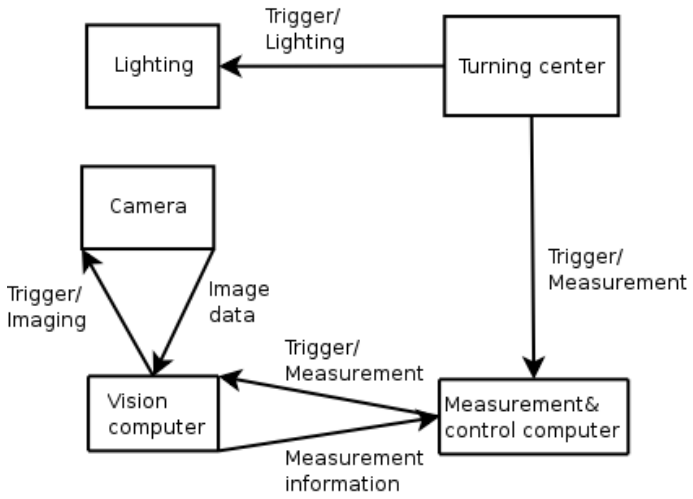


Figure 23: Communication between hardware components.

Components that are connected to the visual measurement system are described in this section. Also the communication details on a hardware level are given.

6.1.1 Turning center

The turning center that the measuring system was built around was Daewoo Puma 2500Y [23]. Main elements that concerned the imaging system were the work-space, *i.e.* where the

work-piece is turned and the inputs and outputs that the turning center could provide to automate the imaging process. The work-space places constraints on the physical location of the camera and lens system. Inputs and outputs of the turning center determined if the imaging system could be automated in a fashion that allowed the system to operate only when the tool is moved to a location where it can be imaged.

During the development of the imaging system, the work-space of the turning center proved to be problematic. The work-space of the Puma 2500Y can be seen in Figure 24. Main components of the work-space are the spindle, which turns the work-piece and the carousel which has twelve tools. The carousel can translate inside the work-space, up to the spindle, and it can also rotate to change the current tool used on the work piece.

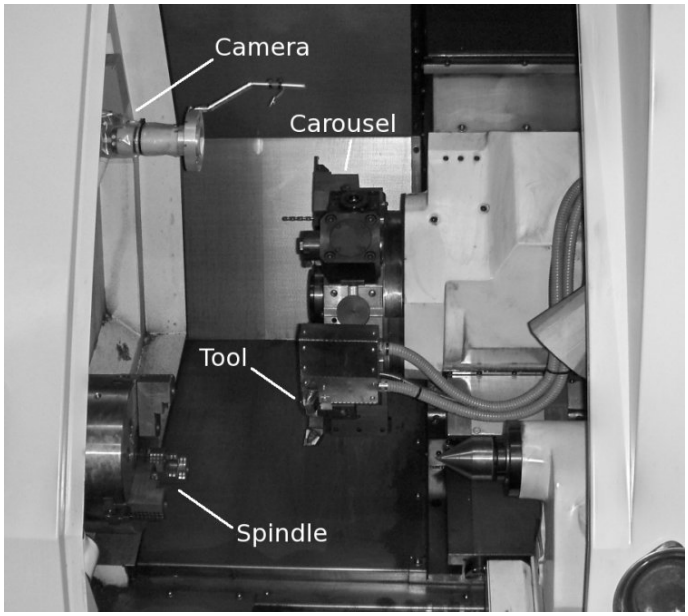
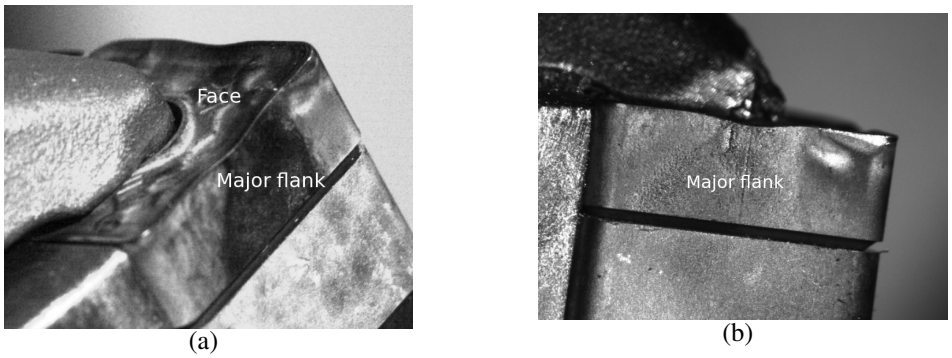


Figure 24: Puma 2500Y work-space

As the carousel can move and rotate, it places constraints on the location of the camera. In this case it was mandatory to place the camera behind the spindle so that the camera could not come in contact with the carousel or its tools at any point. Initially the camera was placed closer to the spindle, so that the taken images contained both the face and major flank of the tool. Image of the tool from this camera position is visible in Figure 25a. However, after the final system components were fitted inside the turning center, it was decided that the camera is too close to the actual cutting process and the camera could be entangled in the continuous metal chips that are generated. The requirement to measure the crater wear was dropped at this point.

The new position was selected so that the major flank would be visible in the camera and the camera would be as far away from the cutting process as possible. Figure 25b show the image captured from final position of the camera. In essence the tool is rotated to the top position of the carousel so that it is in 90 degree angle and the flank face of the tool is parallel to the imaging plane.



(a) (b)

Figure 25: (a) Image from initial position of the camera; (b) Image from the final position of the camera.

Communication with the turning center with the available digital inputs and outputs proved to be difficult. It is possible to read a signal from the outputs when the tool is in the right position for imaging. This is achieved by placing a special code in to the program which controls the turning center. However, it is not possible to signal back through the inputs when the imaging is complete. Instead, a timer based mechanism is available, so that the tool remains in place for a predetermined amount of time. However, timer does not guarantee that the imaging is successful in the alloted time.

6.1.2 Imaging System

The imaging system comprises of following hardware components: a camera, a lens, lighting and a computer for image processing. Specifications for camera, lens and lighting can be found in Appendix 2. The computer is equipped with 2.2 GHz Intel Core 2 Duo processor and 1 GB of RAM (Random Access Memory) with a Linux operating system.

The system specification affected the choosing of the components. Important criteria for selecting the camera was the use of Firewire bus, to minimize wires in the work-space, and that the physical size of the camera should be small. Most important criterion for the camera was the size of pixel array, so that it would fulfill the specified image size and resolution when used with the lens. For the lens, important criteria were working

distance, *i.e.* the distance where the lens is in focus, and the zoom factor of the lens, so that the magnification would be in accordance with the specification.

Specification was not as strict on the lighting. Most important element of lighting was the light which would create the specular reflection, *i.e.* spotlight. For this application, a LED (Light Emitting Diode) was most suitable for its small size and power consumption. Because they do not draw much power, it was also easy to design a controller to drive the LEDs. However, a normal LED would not provide enough illumination at required distance from the tool. This was tested on worn tools before the components were chosen. High-current LEDs (> 350 mA) were chosen instead for the task. The lighting setup is shown in Figure 26.

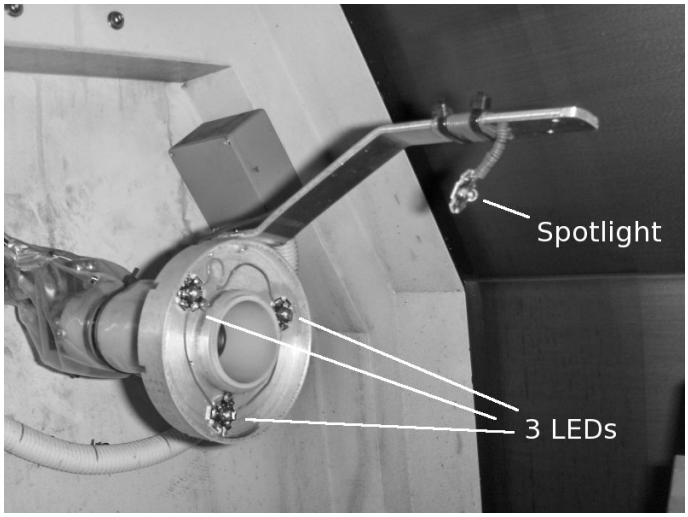


Figure 26: Lighting in place

As explained in Section 6.1.1, the initial position of the camera and the final position of the camera are different. In the initially tested position, the halogen light of the turning center provided enough illumination for the whole tool, except the specular reflection, so that the tool would be easily detected from the image. However, the final position required additional light. It was decided to add three high-current LEDs to illuminate the tool.

The whole imaging system can be seen in Figure 27 inside the turning center. The camera and lens with the lighting were placed on a 3-joint arm, so that the camera could be easily removed from the work-space when needed, for example, when fluids are used during turning. The arm also provided flexibility in moving the camera to an ideal position for imaging the tool. However, the arm is only an intermediate solution in place of a fixed solution. Issues relating to the imaging of the tool will be discussed in Section 6.1.4.

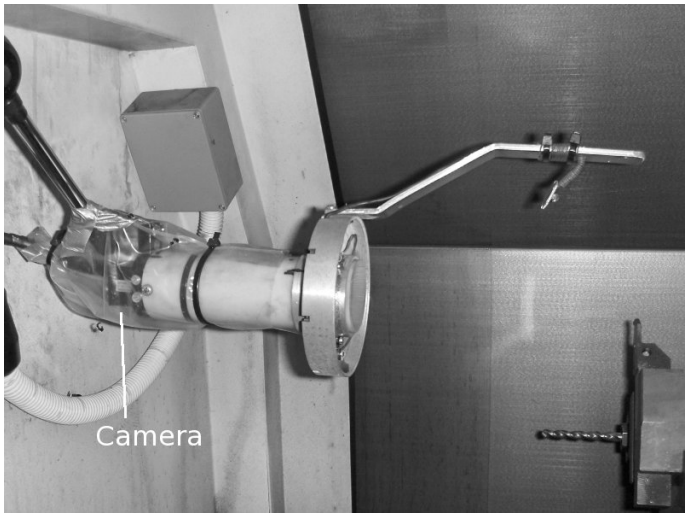


Figure 27: Imaging system in place.

The camera system is protected with a plastic tube. The end of the tube is closed with plexiglass, so that images can be taken, but nothing can enter the tube. This has proved to be an effective solution against the metal chips that are launched from the cutting process to the work-space of the turning center. As the imaging system is placed behind the spindle, possible continuous chips formed during cutting do not affect the camera.

6.1.3 Communication

Specification (Appendix 1) required communication with a remote computer, specifically the measurement and control computer, which handles the cutting process on a higher level than the tool wear measurement system. For this task Ethernet [34] was chosen. It is the most used technique in local area networks as the physical interface between network hosts. It is also possible to form point-to-point connections with Ethernet. This reduces need of hardware as there is no need for additional layers such as routers or switches inside the network between the vision computer and the measurement and control computer. It was also chosen, because most commonly used protocol, with the Ethernet as the physical interface, is TCP/IP (Transmission Control Protocol over Internet Protocol), which was chosen in the specification as the software communication protocol.

Another issue of communication is the switching of the lights on and off during the imaging. As described in Section 6.1.1 the turning center dictates the time when the tool is in proper location for the imaging by signalling. The same signal is used to enable the lights

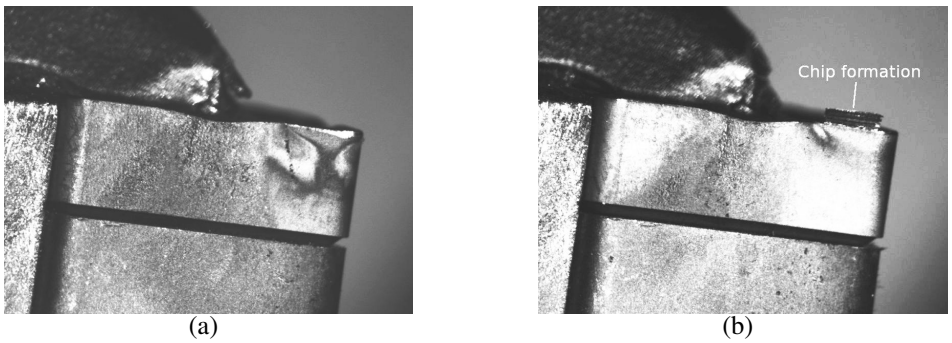
for the duration of the signal. This means that the lights are turned on only when needed, which will prolong the lifetime of the lights and the lights will not distract the operator of the turning center during the actual turning.

6.1.4 Considerations

A number of issues are related to the hardware and physical systems that have to be noted. These affect the hardware functionality as well as the software components.

The first, and the most troublesome, issue is the electrical connection between the imaging system and the turning center. This issue can be fixed quite easily, but the connection between the vision computer and the turning center caused three complete camera failures before the electrical connection was removed by isolating the imaging system from the turning center. The failures occurred, because the Firewire bus of the camera connected the vision computer and the turning center. This formed a large electrical loop that could and did induce current and short-circuited the camera electronics.

Second issue, which is independent of the measurement system, is the formation of metal chips on the tool during turning, which in turn affects the wear measurements. However, this is a random occurrence and affects roughly half of the images taken with the system. This effect can be seen in Figure 28. To minimize the effect on the turning process, an automated system to remove the chip from the tool before imaging is required.



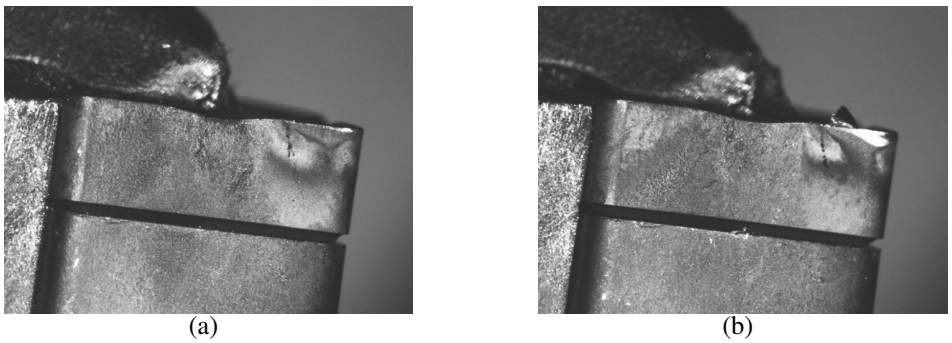
(a) (b)

Figure 28: (a) Clear tool wear zone; (b) Tool wear zone affected by chip formation.

Issues that are related to the system are described next. As the current system is only a prototype, by acknowledging the issues, the system can be improved to give better measurement results.

The current system exhibits a few difficulties when setting the system up for imaging.

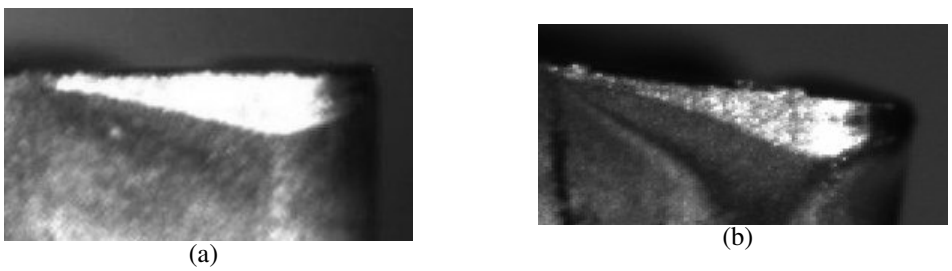
Firstly, the arm of the camera has to be maneuvered to a correct place for experiments as no fixed solution was implemented for this prototype. Secondly, the camera is quite sensitive to vibration due to the arm. This means that the camera must stabilize for a few seconds after the turning center has stopped cutting. Otherwise the image is blurred, which affect the measurements. However, this vibration does not affect the position of the camera, which is illustrated in in Figure 29. The figure shows that the tool has stayed in the image through 118 cuts and multiple hours of operation.



(a) (b)

Figure 29: (a) Image after first cut; (b) Image after 118th cut.

A few issues are also related to the lighting, especially to the spotlight. One problem of the lighting system of the prototype is the need to adjust the spotlight to correct angle in relation of the tool each time experiments are conducted, so that the wear zone would produce specular highlights. This is a key issue, because the software relies on the highlight to measure the wear. However, this is mainly software issue and how the system copes with the variation of the highlight is discussed in Section 6.2 as well as in the Section 4.2 describing the algorithm. Also, as the spotlight is moved between experiments, the quality of the specular highlight changes as well. This is shown in Figure 30. It is recommended that the specular reflection should look like in Figure 30a to get as good results as possible. Again, this issue is related to the measurement algorithm and is discussed in the Section describing the algorithm.



(a) (b)

Figure 30: (a) Good quality specular reflection; (b) Spotty reflection.

6.2 Software components

This section contains detailed descriptions of the software structure and functionality as well as instructions on operating the software. Communication, which is one of the main requirement of the system, is also given a thorough examination. The background for developing the final system through prototyping is given as well.

6.2.1 Software prototype and libraries

To examine the feasibility of the algorithm described in 4.2, a prototype was constructed with Matlab-software, which is a piece of software designed for scientific computing. The prototype was also used as a platform for developing the algorithm. For this purpose, Matlab offers an extensive scripting language with many image processing operations already built in. This sped up the development of the algorithm considerably.

The prototype was developed around the initial test image, shown in Figure 25a. After it was clear that the prototype could detect the wear area in the image as described in the algorithm in Section 4.2, the actual implementation of the final software was started. No other system components, other than the measurement algorithm, were developed in the prototype.

To focus on the problem, *i.e.* the measurement algorithm, a number of software libraries were used during the development of the final software system. The libraries provided essential image manipulation algorithms and image capture from camera, allowing progress to be made on the measuring algorithm.

The most important software library was OpenCV [35], which provided ready-made image manipulation algorithms and its own internal format for images. Image filtering and edge detection are examples of functions that were used from the OpenCV library. A second library, which provided access to Firewire cameras, was internal library developed in the Information Processing laboratory called "itlabcpp". The library provided also an internal image format, which was also used. Common for both of these libraries is that they support C++-language directly (C++ was the chosen implementation language) and they provided functions that would have otherwise taken too much time to implement. So the use of software libraries was well justified.

6.2.2 Overview of the software system

The hardware system was designed after the requirements given in Section 3. The same applies to the software system, as the software is the actual implementation of the requirements, while hardware components act as enablers. The specifications, in Appendix 1, also define some of the requirements for the software design in addition to the hardware requirements.

One of the most important choices in the specification, was the selection of the target operating system. The choice for the implemented system was the Linux operating system with Debian 4.0r1 distribution. With Linux, a number of software libraries were available that would speed up the development.

Another important choice, not defined in the specification, was the programming language. C++-language was chosen for this task, because of its familiarity and the possibility to use C-language software libraries as well as C++-libraries. This is an advantage, especially with Linux, as Linux is heavily based on C-language and the libraries have usually an interface to C-language. C++-language is also an object-oriented language with all the benefits of object-oriented programming.

From the software point of view, problems to be solved were imaging, *i.e.* how to get the image from the camera to software and how to measure the image, and communication with remote computers, so that the imaging can be triggered and measurement results can be sent back to the remote computer. These issues are explored in depth in sections 6.2.3 and 6.2.4.

6.2.3 Software structure

Design of the software system was naturally motivated by the requirements and specification of the whole system. Design was also motivated by the fact that the finished system is still a prototype and as such, should be easily modified to account for changes in requirements during the development. As C++-language was used, the design of the system is also object-oriented.

The high-level design of the system can be seen in Figure 31. This design separates the classes that implement certain functions, such as capturing images or measuring the tool

wear, from each other. This done with the mediator class, Hub, which relays messages between the classes. With the use of interfaces for the functional classes, it is possible to change the classes connecting to the Hub-class without interfering with the whole functionality of the software. This property is used, for example, in testing, where the camera image is taken from a file instead of a real camera.

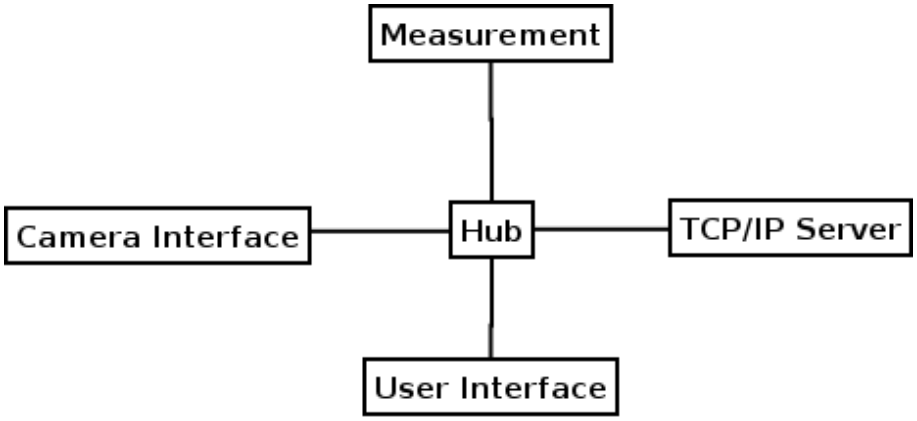


Figure 31: Selected software design.

Another possible design would have been a serial design, shown in Figure 32. In this design, the messages would run from class to class, starting from the trigger coming from the "TCP/IP Server"-class and ending with the transmitted measures through the same class. Compared to the selected design, less classes are needed. However, the problem without the "Hub"-class, is that all the classes need to know the functionality of other classes and the design is more rigid compared to the selected design. Especially problematic would have been the user interface, which would have required interfaces to multiple classes. These issues prevented the selection of the alternative design as modularity was one of the key issues in development.

Complete, but simplified, UML (Unified Modeling Language) class diagram of the software design is found in Appendix 4. The interface classes have the most important methods visible in the diagram. These are the methods that need to be implemented in the implementations of the interfaces.

Most important class in the design is the Hub class. All internal communication between the classes is routed through Hub class. These communication paths are defined in the various interface classes. Hub class also acts as a controller, which controls the measuring process.

CameraInterface class defines the methods for image capturing and settings adjustment.

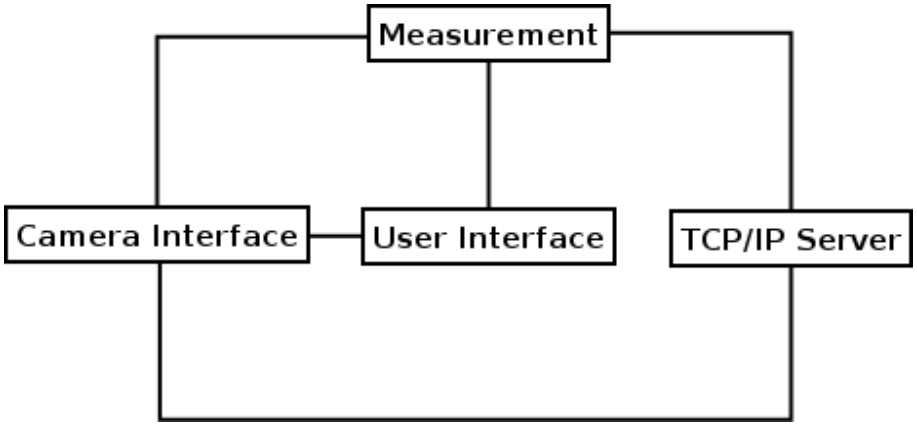


Figure 32: Alternative software design.

Two classes were implemented from the CameraInterface: ImageLoader and FWCameraInterface. ImageLoader class is for testing purposes, it loads an image from the hard drive instead of external source. FWCameraInterface class is used to capture an image from a Firewire camera. By implementing CameraInterface class, it possible to capture image from any source.

MeasurementInterface class has only one implementation, MeasureImage2D. Through the methods in MeasureInterface, it is possible to change the parameters of the measuring process and relay the image to be measured to the class that implements the MeasurementInterface. MeasureImage class implements the algorithm from Section 4.2.

UserInterface-class has methods that allow the software to communicate to the user that is operating the system. It is possible to query the user with a question or a set of questions. UserInterface also allows the user to point things from an image. It is also possible to only show an image to the user. CLIInterface class, which implements these methods, is a command-line user interface with the possibility of showing images.

TCPIPServer class directly implements a TCP/IP-server. It is used for communicating with remote hosts, which allows the measuring process to be triggered. More on communication in Section 6.2.5. NetMsg-class is used as a container for messages communicated between The TCPIPServer and Hub classes.

6.2.4 Software functionality

This section describes the functionality of the software, *i.e.* how the software operates and communicates internally. A flow-chart representing the normal operation is presented in Figure 33. First, the software initializes itself. After initialization, camera alignment is started, if it is required. Camera alignment shows images real-time from the camera as long as user will not give any input. After camera alignment, as well as after intialization, if no alignment was needed, the main process loop starts. The loop waits for commands from the TCP/IP communication channel. If a command is received, the command is first checked that it is a command to initiate measuring. If the command is accepted, an image is taken and measured. The result is then sent back to the remote host using TCP/IP.

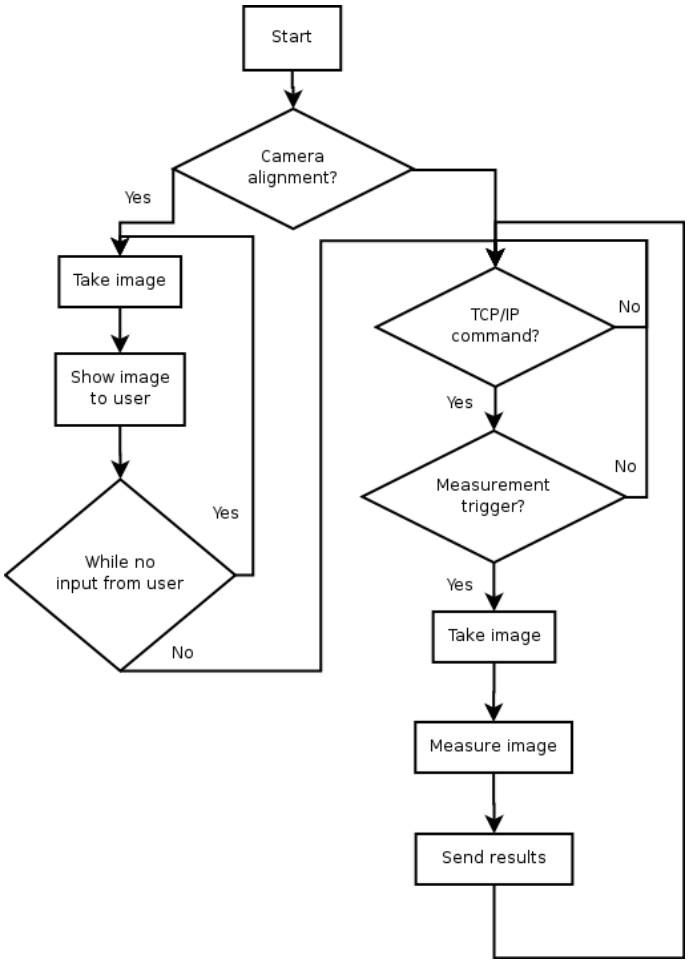


Figure 33: Flow chart of the software operation.

Camera alignment is a functional add-on to the original design. It was not included in the software design until it was noticed that in addition to the measurement, a method for aligning the camera was needed to position the camera to a proper location for the

imaging.

Commands can also be given from the user interface. In this case the user interface replaces the TCP/IP as a communication channel to the imaging and measuring system.

6.2.5 Communication

There are two major communication channels inside the system: Firewire and TCP/IP. Firewire handles the communication from and to the camera which captures the images of tools and TCP/IP is used to communicate with remote hosts, that can request measurement at any time. Of these, Firewire is handled through a specialized software library and TCP/IP communication is handled through normal system libraries.

Whilst communication with cameras (especially industrial cameras) that support Firewire is standardized in IIDC standard [36], communication between software components with TCP/IP can be designed freely. So, a protocol that can both transmit commands to the vision computer and transmit the measurement results back to the remote host had to be designed. The actual implementation of the TCP/IP communication only allows one connection at a time.

The communication protocol is shown in Figure 8 as MSC (Message Sequence Chart). The protocol is based on one command or query that can return multiple answers and the protocol is also transaction based, *i.e.* the connection is closed after each "query-answer" pair. By allowing multiple answers, it is possible to send the measurement results in one message and by ending the results with "END"-message, it is also possible to send less or more data with the knowledge that "END"-message ends the transaction. Each message carries the message name, for instance, "IMAGE", and a parameter value, which allows transfer of new parameter values.

Figure 8 shows two communicating entities, Server and Client. In the software system, Server is the measurement end and Client is the entity which triggers the measurement. All messages between Server and Client consists of one byte characters only. Use of characters ensures that messages are received correctly. Server-Client architecture implements the system specification, as it is possible to trigger the measurement at any time, while the Server is on.

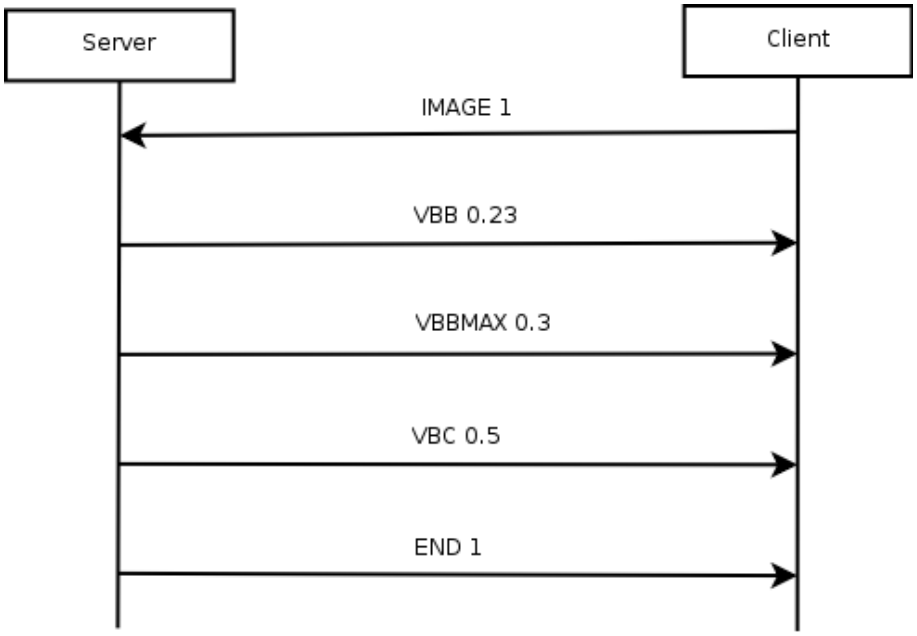


Figure 34: Communication protocol.

6.2.6 Documentation

Documentation of the system, in addition to this thesis, was primarily created with Doxygen [37], which is a source code documentation tool. Doxygen allows the documentation to be embedded directly into the source code. Embedding the documentation eases development and minimizes the time used for documentation. For C++ (implementation language) Doxygen can create automatically a class diagram representing the class structure which visualizes the class design with all available information about the classes and their member variables and methods.

Doxygen requires no other input than the source documentation to create the documentation as Doxygen can parse the language. After this it can create an extensive documentation about the structure of the source code. Doxygen can output the documentation to several different formats, such as HTML.

6.2.7 Parameter settings

The measurement system has a number of parameters for both the measurement algorithm and camera settings. Complete list of the parameters is included in Appendix 5, with

default values and value ranges. In this section the most important parameters that affect the measurement process are examined more closely. Parameter names are typed with `fixed-width font`.

Camera parameters affect the camera settings. This is possible via the Firewire bus. `Camera_Gain` controls the gain of the camera. Gain increases the image brightness but also the noise level, so it is best to keep the gain at low levels but so that measurements can be done. `Camera_Shutter` controls the electronical shutter of the camera. Lower values mean less time is used for each image, which means less light. For this reason, the shutter should be at reasonably high levels, but vibrations disturb the image easier.

Five important parameters are integral to the success of measurement: `Tool_Width`, `Tool_Length`, `Tool_Radius`, `Tool_Depth` and `Tool_Angle`. `Tool_Width` and `Tool_Length` give the vertical and horizontal edge lengths respectively in millimeters. `Tool_Width` and `Tool_Length` affect the calibration and, thus, are critical for successful measurement results. `Tool_Radius` and `Tool_Depth` do not affect calibration, but they determine the wear zones (C, B). `Tool_Radius` informs measuring algorithm of the tool nose radius and `Tool_Depth` the used cutting depth. `Tool_Angle` describes the angle in degrees between the imaging plane and tool. The angle, α, is seen in Figure 19. This angle has an effect on the wear zone placement in the image.

A number of parameters control the preprocessing of the image. `Pre_Bright` controls the brightness adjustment and `Pre_Contrast` adjusts the contrast. If lighting conditions are changed compared to the experimental setup, these parameters might need a change. `Canny_Low` and `Canny_High` set the low and high thresholds for the Canny edge detector described in Section 4.2.3. Changing these might also be necessary if lighting conditions are changed.

Detecting the lines is done through the Hough transform. The only relevant parameters are `Hough_Vert`, `Hough_Horiz` and `Hough_Search`. `Hough_Vert` and `Hough_H-oriz` give the angle (in degrees) of the vertical and horizontal edge of the tool, as described in this thesis earlier. Angles do not have to be horizontal or vertical. `Hough_Sea-rch` describes the search space (in degrees) for the edges in Hough space. The given number is the total search area, *i.e.* `Hough_Search` value is divided into searching lower and higher than given angles. If the tool is kept at approximately same angle as in the example images, there is no need for change.

Actual measurement is based on a number of parameters. `Max_Wear`$\cdot 0.95$ determines the maximum measured wear in millimeters. The last 5 % is used for detecting outliers, *i.e.* samples that are measured wrong. As the wear measurement is based on median differences, two important parameters are `Median_Size` and `Median_Diff`. `Median_Size` determines the size of the vector of pixels that is used to calculate median for each pixel and `Median_Diff` is used as a threshold between the median values to decide whether the pixel is at wear edge. `Filter_Size` determines the size of the Gaussian filter used before measuring. This parameter is quite important when the specular reflection from the tool wear is poor. However increasing the filter size may cause measurement failure, as the wear edge is not as sharp as without filtering. `Start_Offset` determines the actual starting point from the tool edge in pixels. This parameter can be used to disregard the non-reflecting part of the tool wear at the tool nose.

Automatic calibration can be overridden with `Calibration_Override`. If calibration is overridden four values must be known, `Calibration_Width`, `Calibration_Length`, `Calibration_ToolX` and `Calibration_ToolY`. `Calibration_Width` and `Calibration_Length` correspond to the tool width and length but in pixels instead of millimeters. `Calibration_ToolX` and `Calibration_ToolY` give the tool nose position as pixels in the image.

7 EXPERIMENTS AND DISCUSSION

This section describes the experimental setup and the results of the experiments made with both the visual measurement system as well as the wear modelling, *i.e.* the prediction of the wear. The setup for all experiments, including the visual measurement system and prediction experiments, is shown in table 1.

Table 1: Setup for experiments.

Tool	SNMM 120412-PR
Type	GC 4015 (ISO P15)
Manufacturer	Sandvik Coromant
K_r	$75°$
ϵ_r (nose radius)	$90°$
λ_s (inclination)	$-6°$
γ (chipping angle)	$-6°$
Work-piece	34CrNiMo6
Diameter	$140\ mm$
Length	$381\ mm$

7.1 Visual measurement system

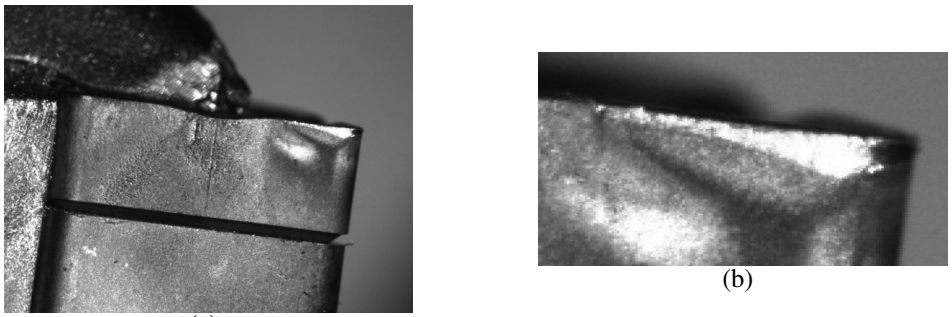
The developed measurement system and its performance was tested with both synthetic tests and application tests. Data for synthetic tests was generated with computer and may not reflect the real results that the algorithm may generate. However, synthetic tests are a useful measure of specific phenomena that may appear in real images. By generating the data by computer, the changes in images can be controlled precisely, which is hard to achieve with data captured from the real environment. On the other hand, data for application tests was captured during turning process and as such represent the final system environment and its phenomena in full.

7.1.1 Synthetic tests

Total of five synthetic tests were developed to measure the effectiveness and performance of the measuring algorithm. The single image used in every synthetic test is shown in Figure 35a. The calibration was made by hand for the synthetic tests, so the image in Figure 35b was used in majority of tests as only the measurement performance was tested,

not the calibration performance. Each image was normalized, so that each channel of the RGB image had values in the interval $[0, 1]$. Measurement results for the unaltered test image were

$$\begin{aligned} VB_B &= 0.22mm, \\ VB_Bmax &= 0.36mm, \\ VB_C &= 0.40mm. \end{aligned}$$



(a) (b)

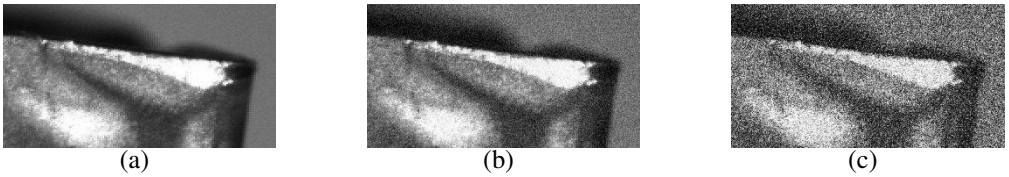
Figure 35: (a) Complete tool image; (b) Wear area of the tool.

Two tests evaluate the algorithms ability to handle common problems caused by camera and lens, which are camera noise and that the lens of the camera is set slightly out of focus. Camera noise is present almost always and is dependent on the lighting and the gain of digital cameras. If the camera lens is out of focus, the lens can be adjusted manually, but especially in the application of this thesis, the depth of field can be only few millimeters, which results in inevitable blurring of the tool image at some points. Base image for the tests is visible in Figure 35b.

Three tests were devised to test how the environment effects the measurement algorithm. These tests measure effect of global lighting changes, effect of position the tool and effect of orientation of the tool. Image in in Figure 35b was used in the global lighting test, while image in Figure 35a was used in tool position and orientation tests to generate more images of the wear area.

Camera noise was simulated by introducing additive noise in to the test image. The noise is Gaussian noise with zero mean and variable variance. Noise was added to each of the three channels of the RGB image. As an example, Figure 36 shows the test image with three different levels of Gaussian noise. In total 14 levels of noise were used, in interval

$[10^{-6}, 0.001]$ with one order of magnitude increments and then in interval $[0.01, 0.1]$ with 0.01 increments. 20 test images were generated for each noise level, in total 280 sample images, with 80 samples in first interval and 200 samples on the second interval.



(a) (b) (c)

Figure 36: Images with added Gaussian noise: (a) Noise variance 10^{-6}; (b) Noise variance 0.01; (c) Noise variance 0.09.

Figure 37 presents the measurements for each test image. Measurements for VB_B, VB_B max. and VB_C are in separate subfigures. Error bar plot in Figure 38 shows the bias of the measurements by plotting the average over the 20 test samples in each noise level. Error bars plot variance over the 20 test samples. Figure 39 plots MSE (Mean Squared Error) of each measurement over each noise level.

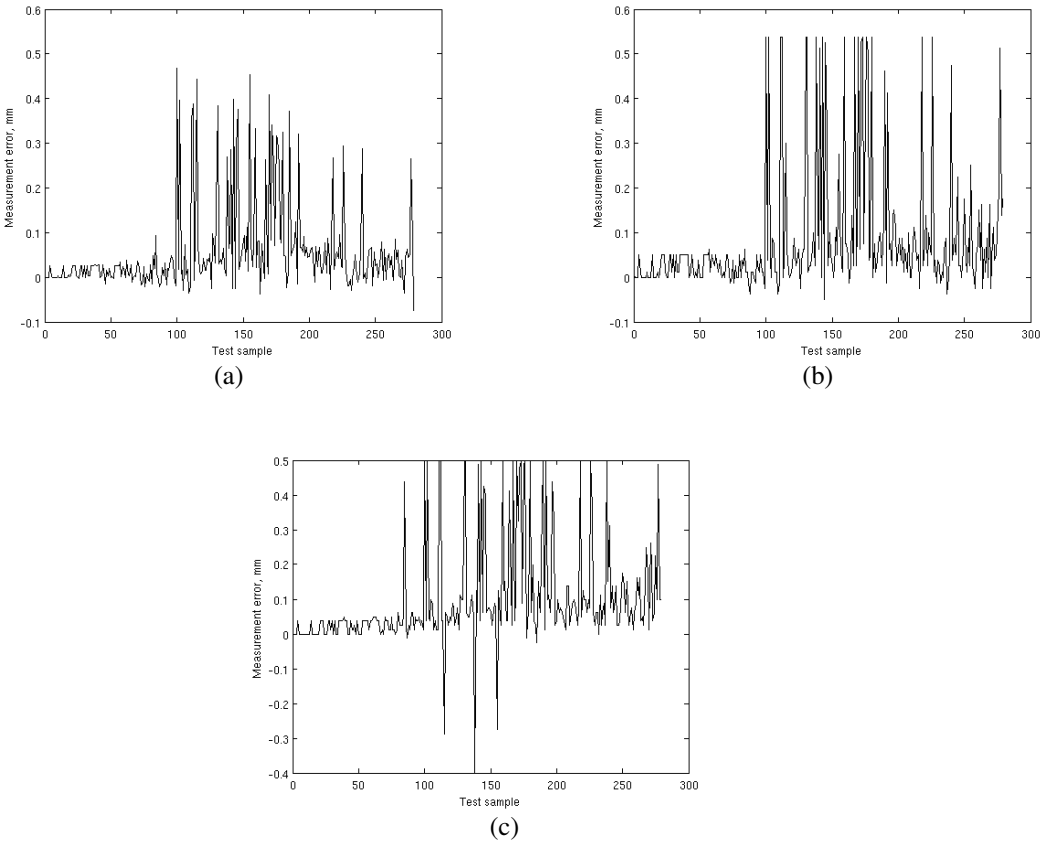


Figure 37: Measurements under noise: (a) VB_B; (b) VB_B max; (c) VB_C.

From the measurement results seen in figures 37, 38 and 39, it is clear that the mea-

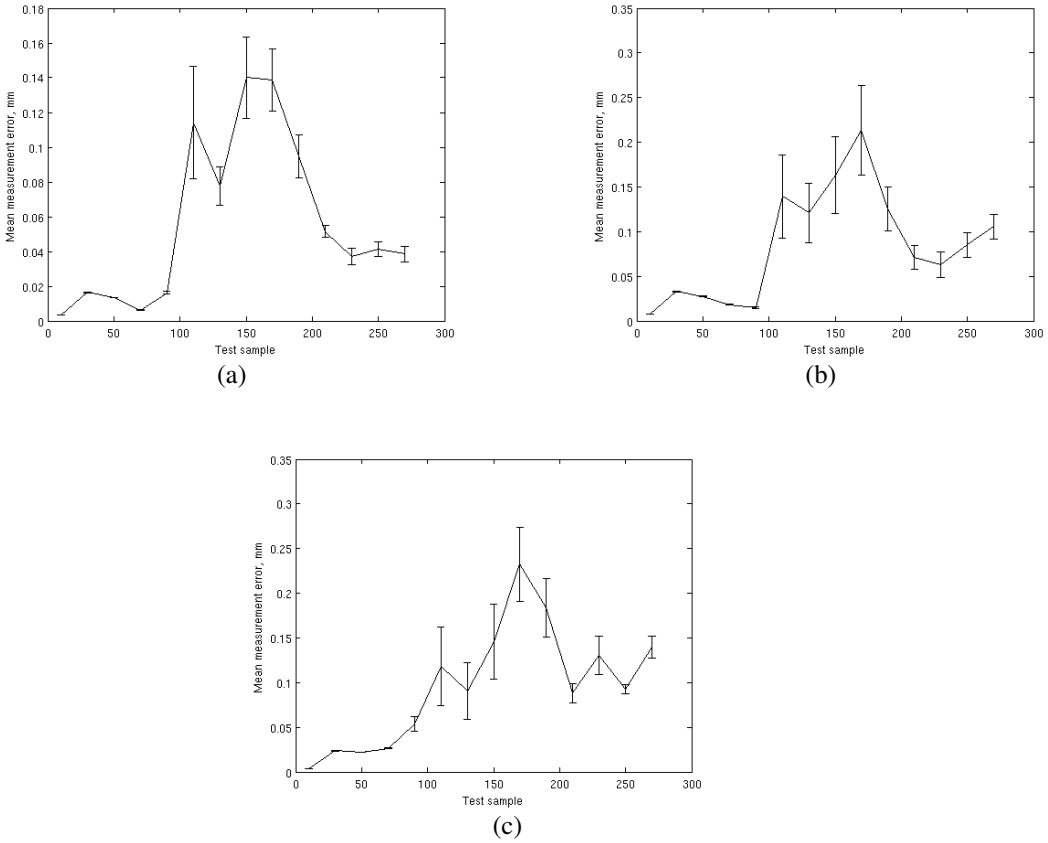


Figure 38: Noise inflicted bias and variance: (a) VB_B; (b) VB_B max; (c) VB_C.

surement algorithm can handle some noise. The variance of the results starts to grow considerably when the Gaussian noise variance grows beyond 0.01, seen in Figure 36b. This means that the reliability of the measurements is poor after this threshold. The noise at this level starts to affect the tool edges in the image, which results in poor detection of the tool tip and can lead to arbitrary measurements.

The out of focus effect was simulated by a Gaussian filter with 5×5 mask. Example images are shown in Figure 40. Deviation of the Gaussian filter was in interval $[0.01, 1]$ with 0.01 increments and in interval $[1, 9.9]$ with 0.1 increments. This resulted in 189 test images, with 99 test images in the first interval and 90 on the second interval.

Measurement results for the test images filtered with Gaussian filter are shown in Figure 41. Figure 42 has the bias and variance of the measurements. Bias and the variance is calculated over sets of 10 samples. This also applies to calculation of MSE, shown in Figure 43.

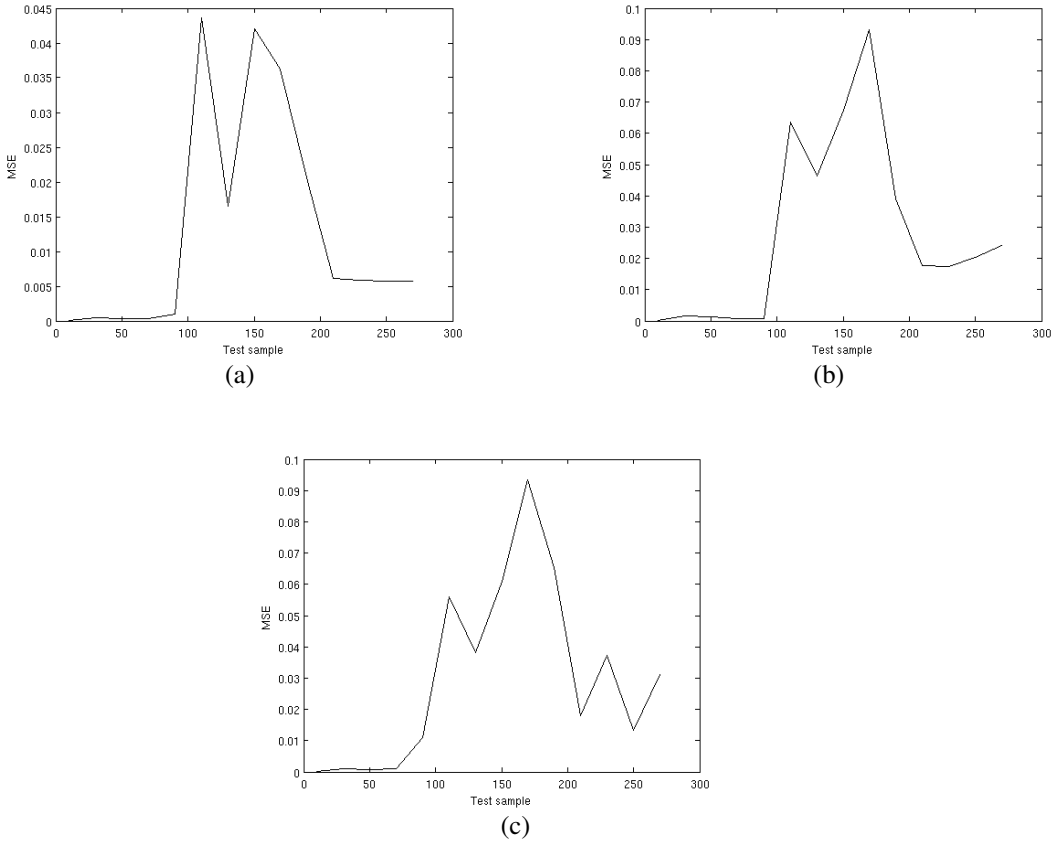


Figure 39: Measurement MSE under noise: (a) VB_B; (b) VB_B max; (c) VB_C.

From the measurement results in figures 41, 42 and 43, blurring of the image does not affect the results nearly as much as the additive noise. This is expected, as the algorithm itself relies on Gaussian filter to smooth the image so that the wear area is more uniform. Also the algorithm uses contrast and brightness adjustment to extract hard edges from the image, so the effect of the Gaussian filter is somewhat cancelled at that point.

Changing global lighting conditions was tested by changing the overall brightness of the test image in interval $[0.5, 1.5]$ in 0.01 increments, total of 101 test images. All values were clamped to interval $[0, 1]$. Example images can be seen in Figure 44.

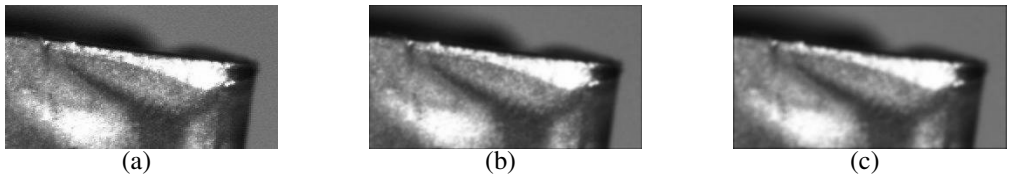


Figure 40: Images filtered with Gaussian filter: (a) Deviation 0.01; (b) Deviation 1.0; (c) Deviation 9.9.

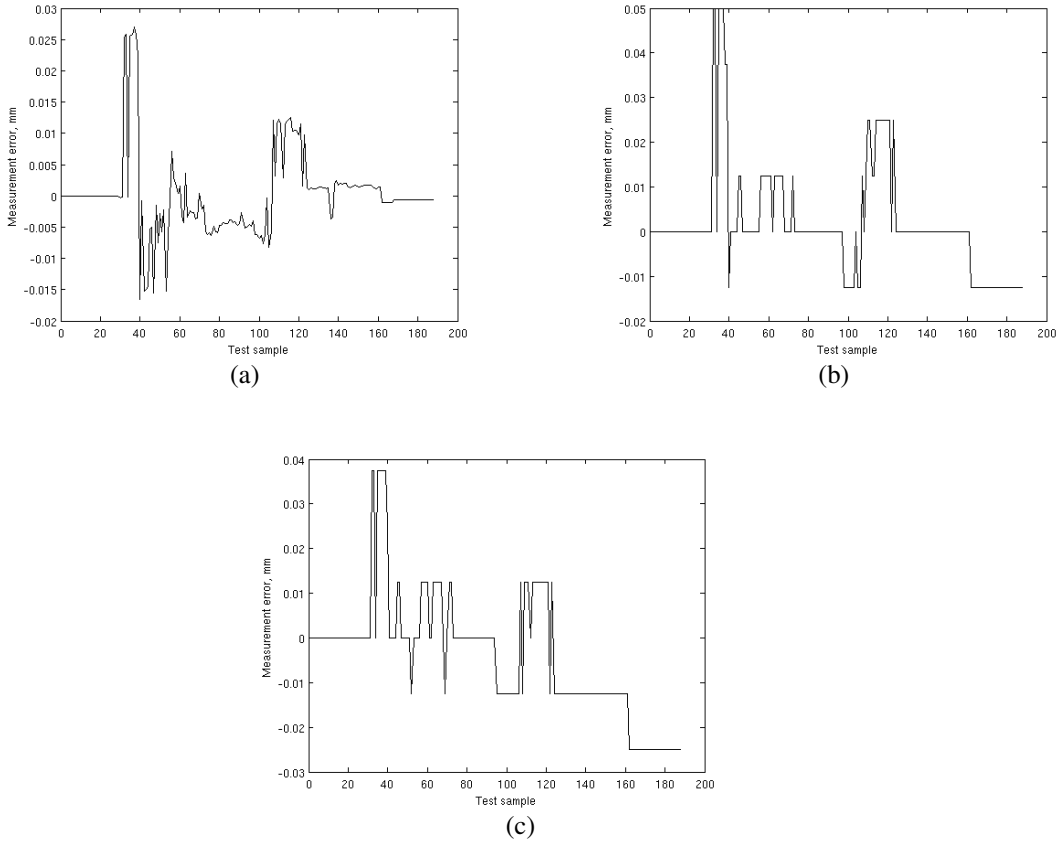


Figure 41: Measurement under blur: (a) VB_B; (b) VB_B max; (c) VB_C.

Measurement error for the test images with image brightness variation is shown in Figure 45. Figure 46 has the bias and variance of the measurement error. The bias and the variance are calculated over sets of 10 samples as with the Gaussian filter test. The MSE is also determined over 10 samples, the result is shown in Figure 46. The large increase of error in the last samples is naturally caused by the image brightness increasing to a limit where the algorithm cannot determine the boundary of the wear due to the global brightness, which is near the brightness of the specular reflection.

Effects of position of the tool on measurements was tested by generating a set of 20 images of the tool nose from the image shown in Figure 35a. Tool position in the test images was randomized. Error in measurements can be seen in Figure 48. Effect of tool rotation on the measurements were tested by rotating the image in Figure 35a in interval $[-25, 25]$ in 5 degree increments. The images were then cropped manually to include only the tool nose. Figure 48 represents the error in measurements.

Tables 2- 4 present error measures for VB_B, VB_B max. and VB_C. Large maximum

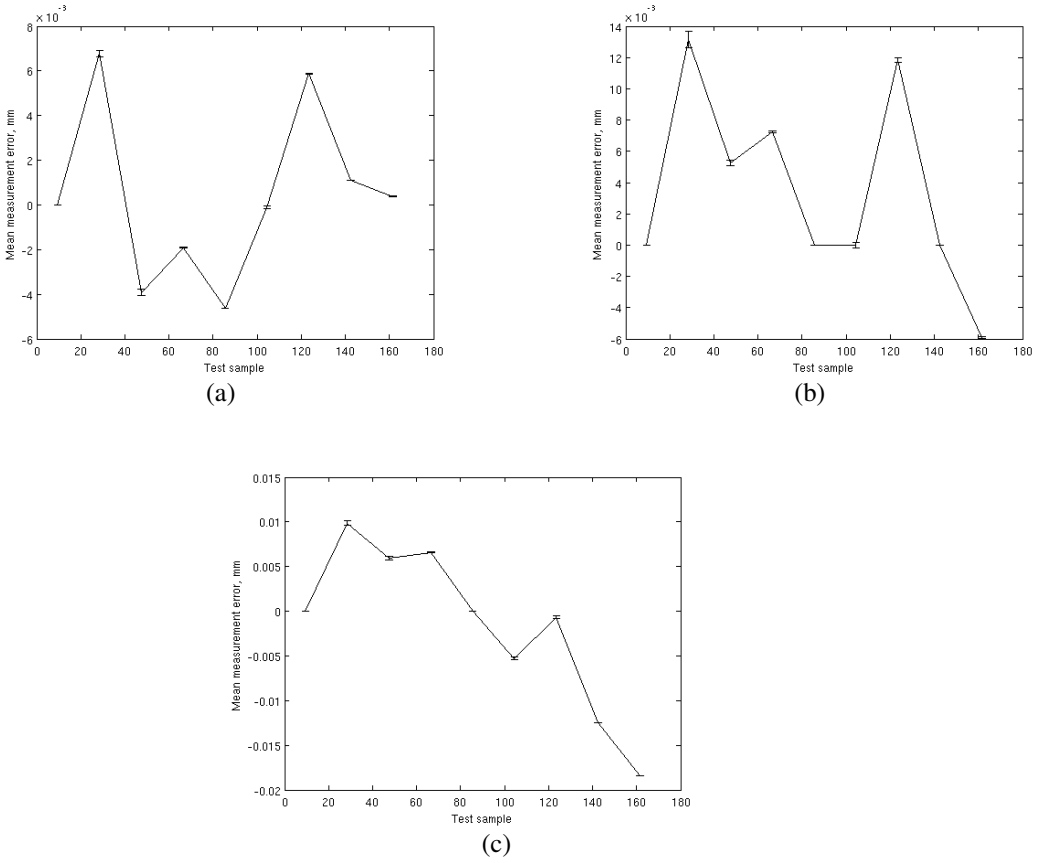


Figure 42: Blur inflicted bias and variance: (a) VB_B; (b) VB_B max; (c) VB_C.

error in rotation can be attributed in part to the fact that rotating the image changes pixel neighbourhoods slightly, which in turn, causes the algorithm to detect the vertical edge fo the tool at wrong position. Otherwise, the average error and error variance are small compared to the real measurement. Position error is small in every category, which can be seen from the Figure 48. Overall, the error in both position and rotation is caused by small errors in detecting the position and rotation of the tool as the Hough accumulator can only represent a discrete number of position and angles of the lines.

Table 2: Error in VB_B

	Position	Rotation
max. abs. error	0.02128	0.0707
mean error	0.0100	-0.0161
error var.	$0.1287 \cdot 10^{-3}$	$0.5050 \cdot 10^{-3}$
MSE	$0.1223 \cdot 10^{-3}$	0.0007

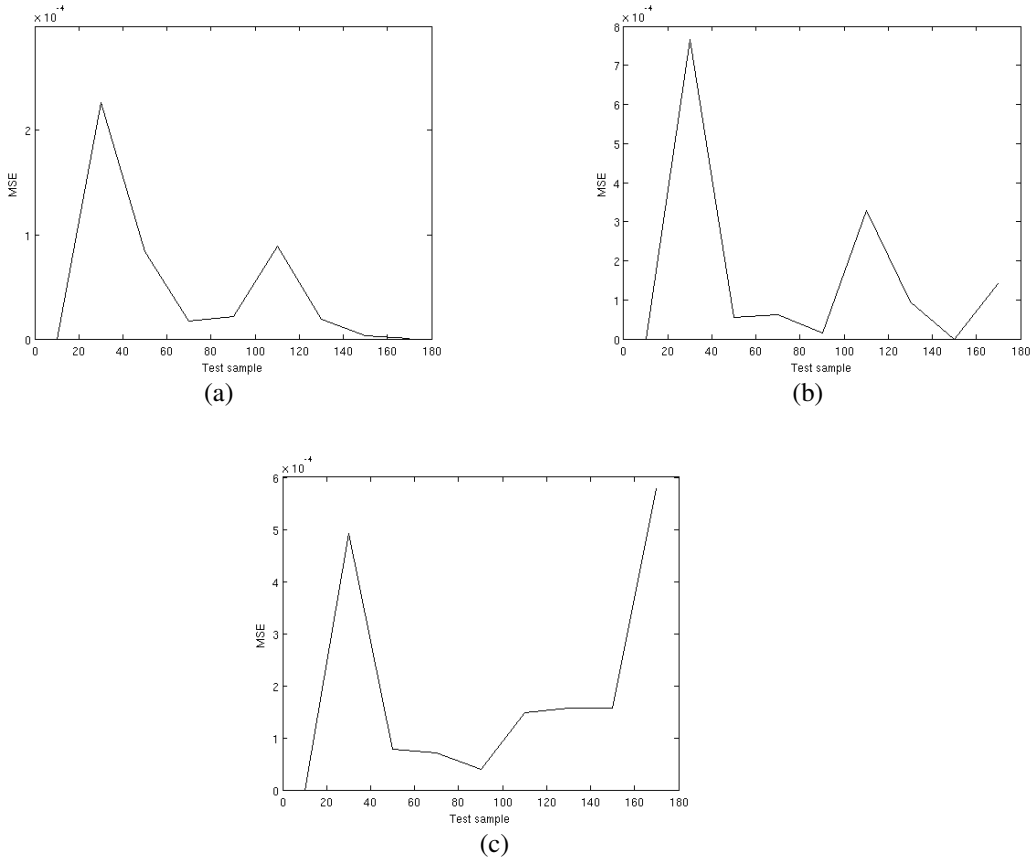


Figure 43: Measurement MSE under blur (a) VB_B; (b) VB_B max; (c) VB_C.

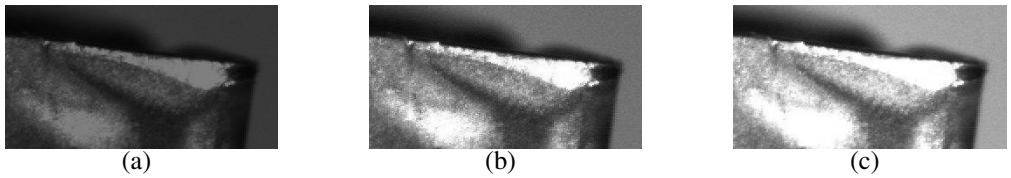


Figure 44: (a) Brightness 50%; (b) Brightness 100%; (c) Brightness 150%.

7.1.2 Application tests

Application tests measure the performance of the flank measuring algorithm in the actual working environment. The tests also measure the algorithm against human measurements.

Image data was collected by the measurement system during turning. 32 images were captured that represent the entire tool life from a new tool to completely worn out tool. Simultaneously, flank wear was measured manually by two experienced measurers. Measurements were kept separated, so that the measurers would not have effect on each others results. 24 results were collected out of the 32 captured images.

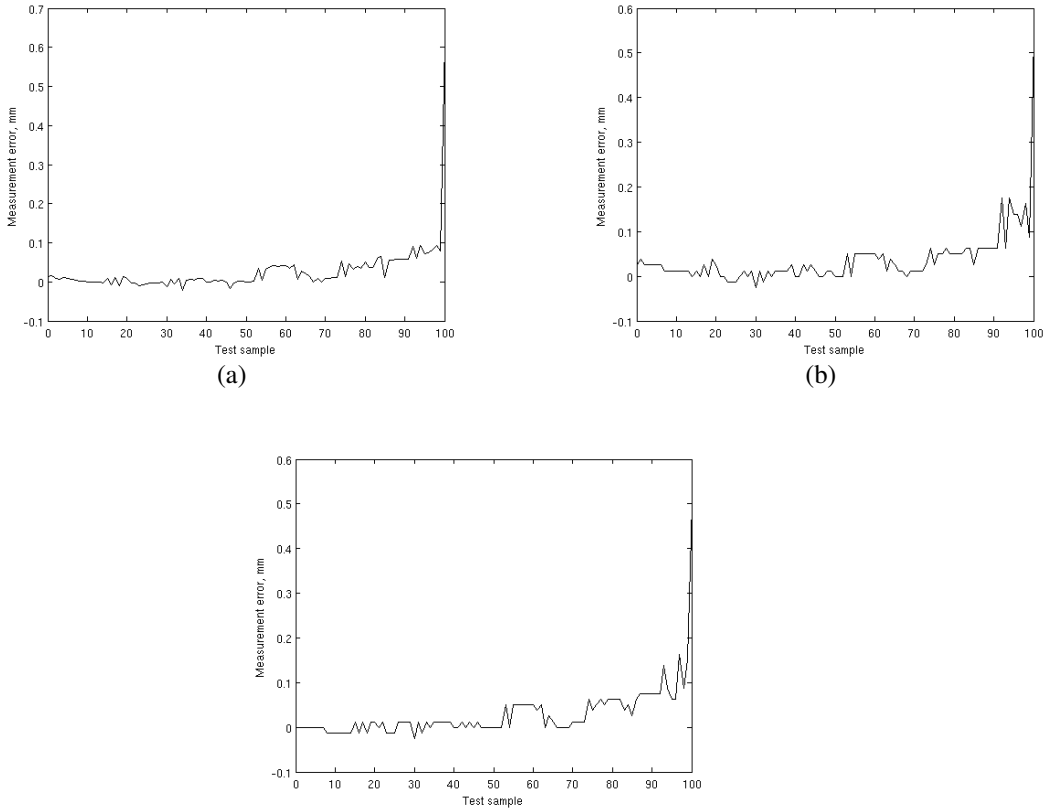


Figure 45: Measurement error under brightness variation (a) VB_B; (b) VB_B max; (c) VB_C.

Some images had to be removed from the captured test set because of the chip formation explained in Section 6.1.4. Also three images had to be removed from the start of the test set as they did not exhibit enough wear for the system to work reliably. A total of 14 images were left after with measurement results from the manual measurements. All the results are based on this set 14 images.

Comparison of the measurement system to the manual measurements is shown in Figure 50. Figure represents the average manual measurements and the measurements of the

Table 3: Error in VB_B max

	Position	Rotation
max. abs. error	0.0375	0.0875
mean error	0.0002	-0.0182
error var.	$0.2401 \cdot 10^{-3}$	$0.8864 \cdot 10^{-3}$
MSE	$0.3281 \cdot 10^{-3}$	0.0011

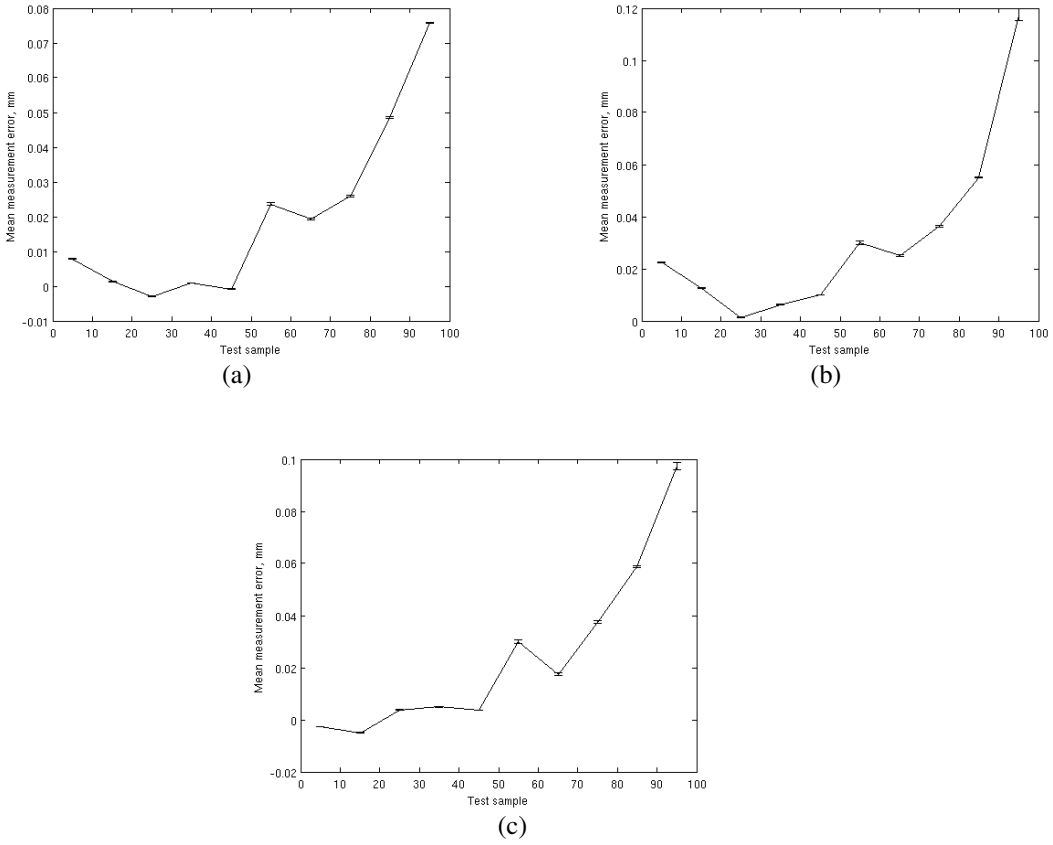


Figure 46: Measurement error bias and variance under brightness variation: (a) VB_B; (b) VB_B max; (c) VB_C.

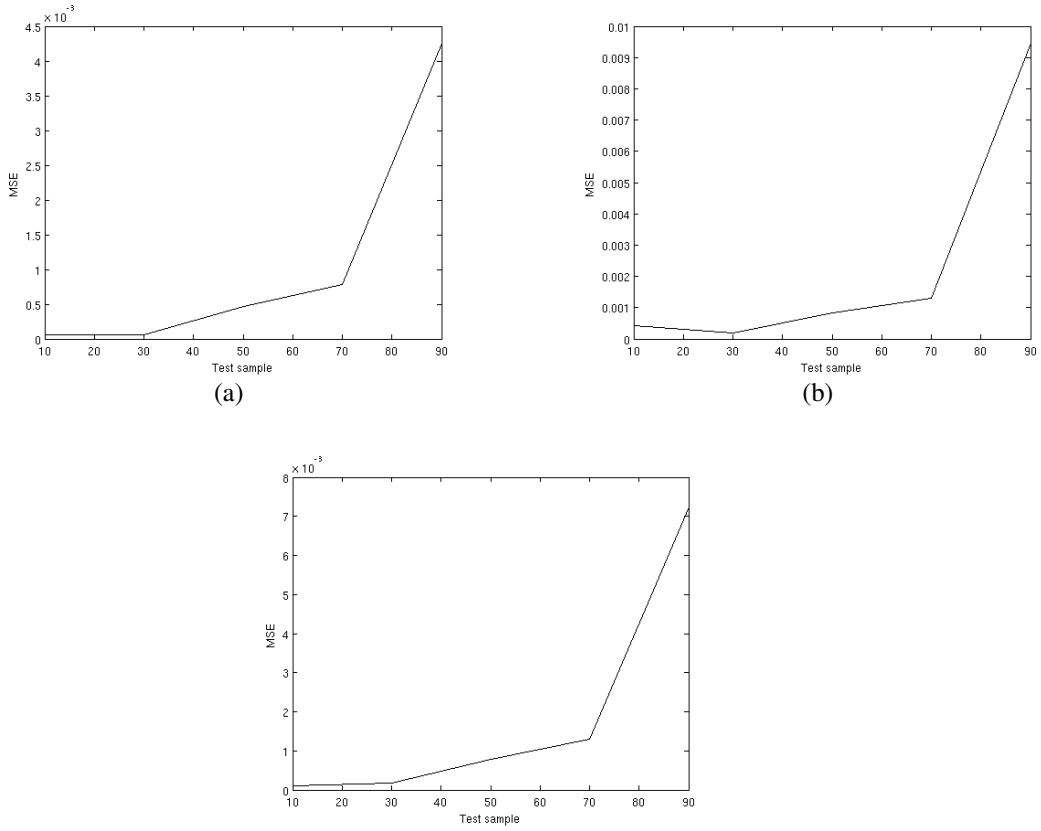


Figure 47: MSE under brightness variation: (a) VB_B; (b) VB_B max; (c) VB_C.

measurement system. Error bars indicate one standard deviation of the manual measurements.

As Figure 50 shows, the measurement algorithm has some difficulties at the start and at the end of tool life. Peaks formed at the start are a result of measurement failure. Main reason for these failures is that the flank wear is not yet large enough to strongly reflect the spotlight. At the end of the tool life the error also grows, but the deviation of the manual measurements also grows, which means that the measurement results fall within acceptable range.

Table 4: Error in VB_C

	Position	Rotation
max. abs. error	0.025	0.05
mean error	0.0037	-0.0205
error var.	$0.2484 \cdot 10^{-3}$	$0.3210 \cdot 10^{-3}$
MSE	$0.2500 \cdot 10^{-3}$	0.0007

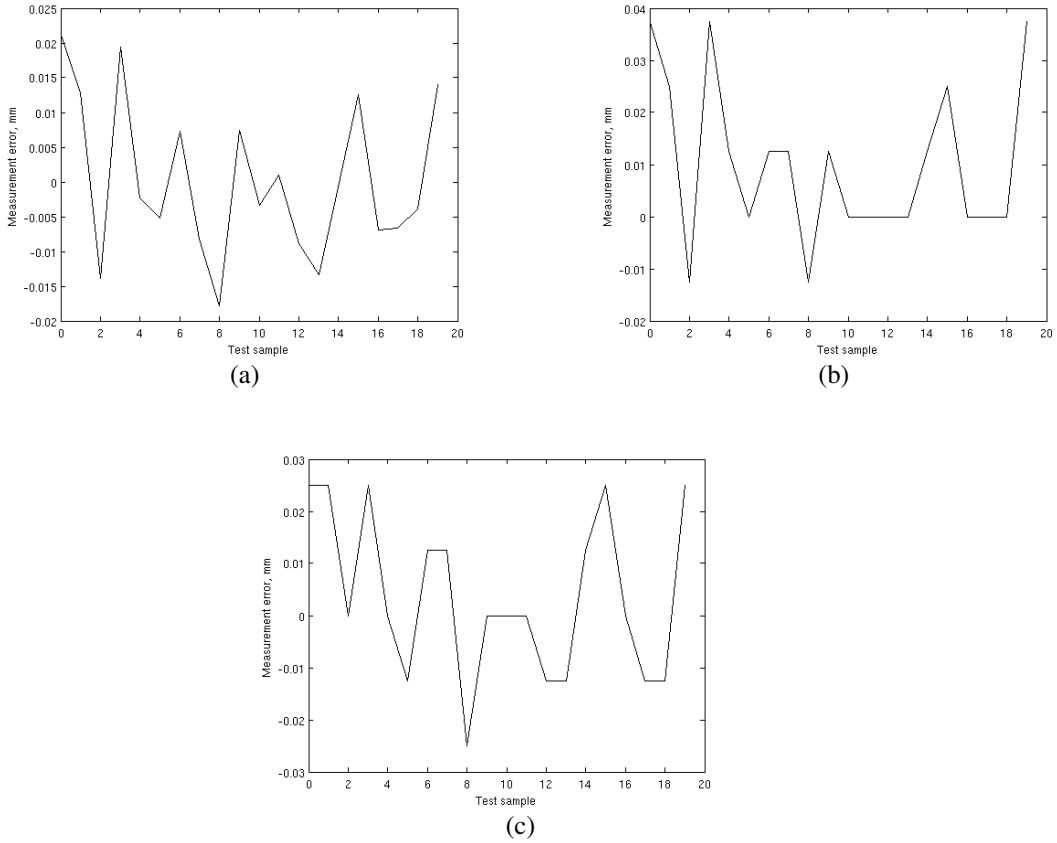


Figure 48: Measurement error with random tool position: (a) VB_B; (b) VB_B max; (c) VB_C.

Figure 51 shows the error as histograms, which are divided into 7 bins. The error peaks visible in Figure 50 can be seen in the histograms as single large errors. Main conclusion from the histograms is that the error is quite small ($< 0.05\ mm$) in most cases.

Figure 52 shows the error of the measurements compared to the average manual measurements. The figure also shows the percentage of the error compared to the manual measurement. The error peaks are visible, just as in Figure 50. Error measures for the measurement system are presented in tables 5-7. Error measures have been calculated with failed measurements as well as without the failures to indicate how well the system performs without failures. Removing the failures leaves 12 samples to the test set.

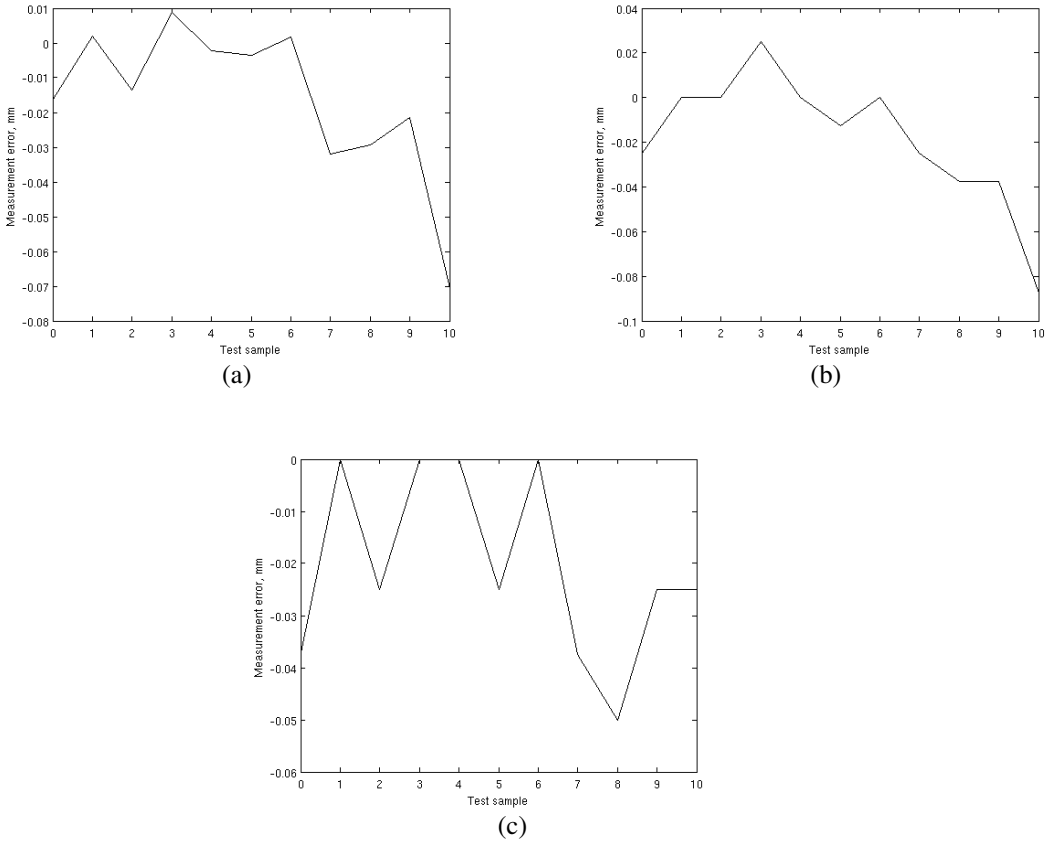


Figure 49: Measurement error with rotated tool image: (a) VB_B; (b) VB_B max; (c) VB_C.

7.2 Modelling experiments

The aim for the modeling and prediction experiments was to find out the best method and best data that could model and predict the wear behaviour. Experiments rely on the methods described in Section 5. VB_B is the only wear measurement used in modeling. This can be justified by the experiments in Section 7.1, which show that the measurements for VB_B, VB_B max. and VB_C behave similarly. Another feature used throughout the modeling is that the modelled variable is the flank wear speed, not the flank wear directly, which was also discussed in Section 5.

Table 5: Measurement error in VB_B

	With failed measurements	Without failed measurements
max. abs. error	0.1888	0.1888
max. per. error	75.5556%	59.9365%
mean error	0.0177	0.0152
MSE	0.0044	0.0047

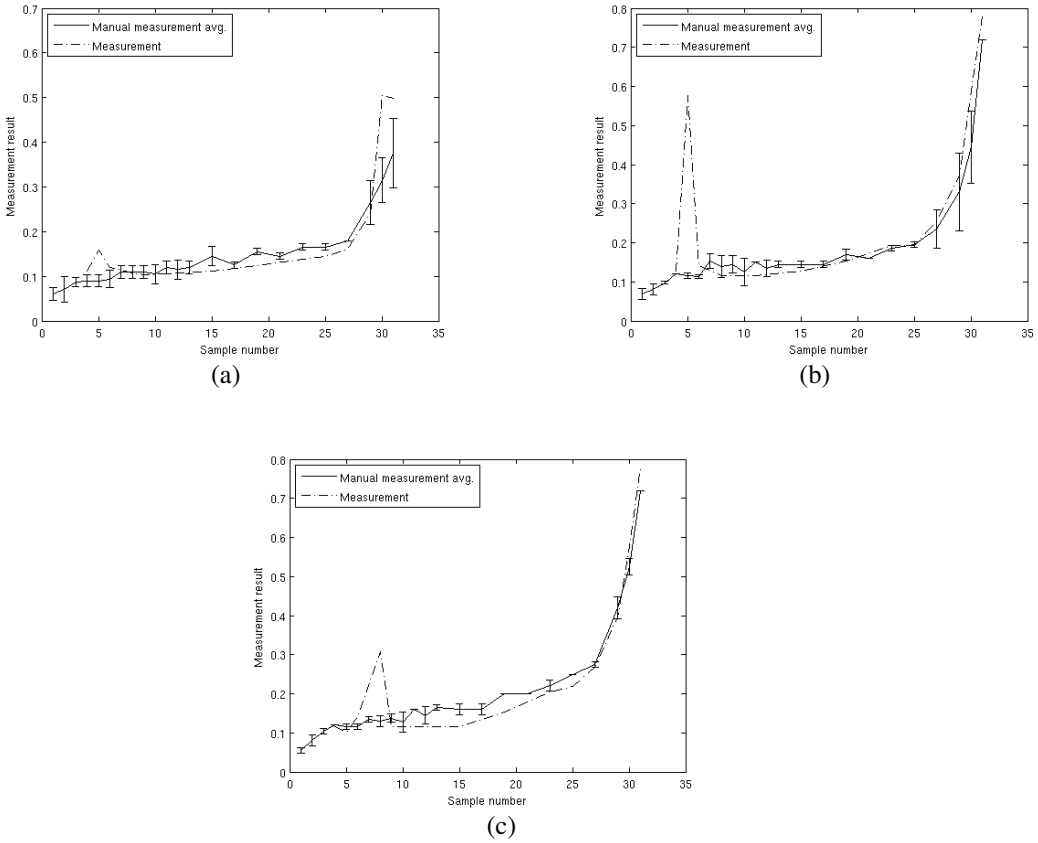


Figure 50: Measurement comparison: (a) VB_B; (b) VB_B max; (c) VB_C.

7.2.1 Wear experiments

To be able to use the modeling methods, data had to be collected of the flank wear behaviour. A series of 12 wear experiments were conducted. The setup for the experiments is shown in Table 8. However for the purpose of testing the methods introduced in Section 5, only the first of the wear experiments in Table 8 will be used in modelling experiments.

The data gathered from the first five experiments is shown in table 9. Time and wear

Table 6: Measurement error in VB_B max

	With failed measurements	Without failed measurements
max. abs. error	0.4610	0.1438
max. per. error	400.8696%	32.3146%
mean error	0.0461	0.0174
MSE	0.0174	0.0025

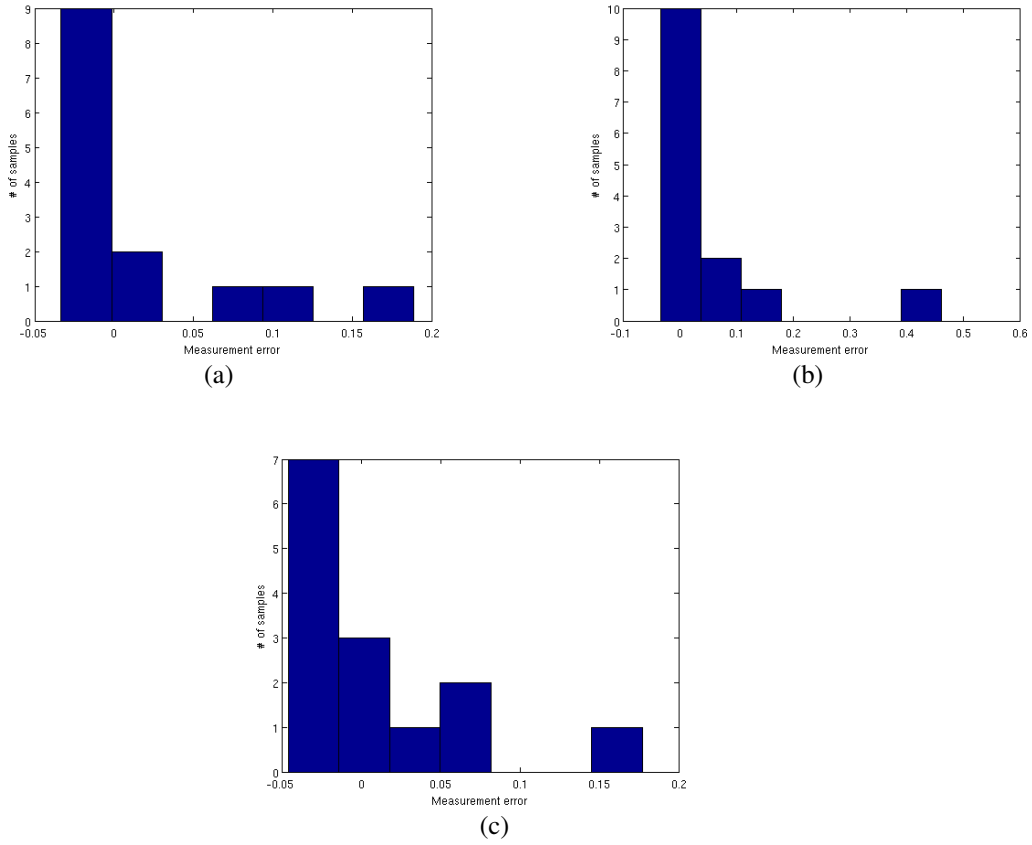


Figure 51: Measurement error histograms: (a) VB_B; (b) VB_B max; (c) VB_C.

measurements indicate the final machined time and the final wear. After each cut, image was captured of the tool flank wear. During each cut, data was collected, including acceleration and forces affecting the tool and the temperature of the tool holder. Features that were used in the modeling were the means of each individual data type. So, for each cut, only one feature vector was extracted.

Of the five tests in table 9, tests 2-5 will be used as training sets and test 1 as a test set. The first wear experiment was conducted so that the manual measurements of the flank wear were taken after each cut. This led to a much longer lifetime of the tool. Tests 2-5

Table 7: Measurement error in VB_C

	With failed measurements	Without failed measurements
max. abs. error	0.1772	0.0608
max. per. error	136.3077%	22.4348%
mean error	0.0044	−0.0085
MSE	0.0034	0.0013

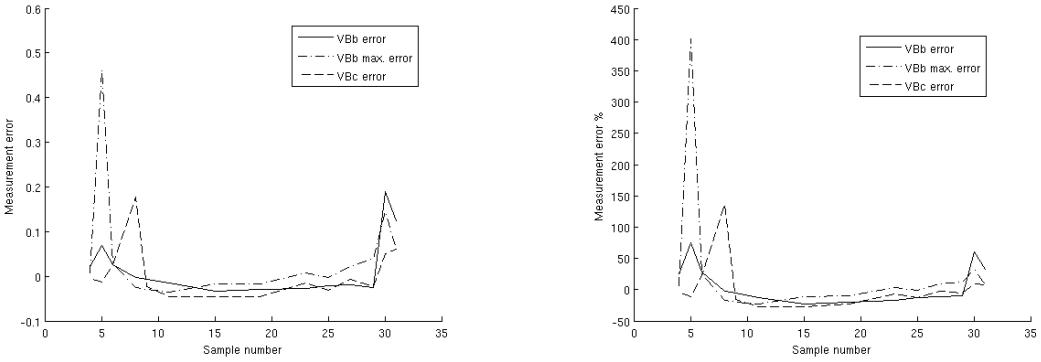


Figure 52: Measurement error: (a) Measurement error; (b) Measurement error in percentage.

Table 8: Wear experiments

$V\ (m/min)$	$f\ (mm/rev)$	$b\ (mm)$	# of experiments
145	0.4	3	5
175	0.7	3	5
175	0.4	3	1
145	0.7	3	1

were however performed similar to each other.

For all the tests, flank wear speed was calculated for each sample over a 5 sample neighbourhood using weighted least squares line fit in wear-time-coordinates. Because the flank wear was measured partly by the vision system and partly by hand, there were some noise in the wear measurements, which caused negative flank wear speeds. These samples were removed from the sets.

From the data gathered during turning, only one type of data were found to correlate strongly with the wear, which was feed force. This feed force was used as extra measurement in addition to the flank wear in modeling. Feed force is indicated in the modeling as

Table 9: Collected data

# of exp.	# of cuts	time	VB_B	VB_B max	VB_C
1	32	1731.8	1.01	1.35	1.38
2	24	1282.3	0.34	0.56	0.56
3	27	1430.9	0.41	0.62	0.63
4	24	1268.1	0.41	0.59	0.62
5	23	1220.5	0.62	0.84	0.87

force only.

7.2.2 Linear regression

Linear regression was tested with three different regressor combinations: only wear, wear with force and only force. The observation was flank wear speed in the regression. The results as MSE are shown in table 10. All data was scaled to $[0, 1]$ and MSE was calculated on this scale as well.

Table 10: Linear regression MSE

	Test set	Training set
Wear	0.0048	0.0020
Wear+Force	0.0048	0.0019
Force	0.0556	0.0045

As can be seen in table 10, using wear or wear and force is better than just using force as a regressor. However, there is not a large difference in using only wear rather than wear and force with linear regression.

7.2.3 Artificial neural networks

Artificial neural networks were tested the same way as the linear regression with all three combinations and all data was scaled to $[0, 1]$. The used artificial neural network, feedforward neural network, was created and trained in Matlab's neural network toolbox. ANN was trained with backpropagation using Levenberg-Marquardt algorithm. Early stopping was used, so that the ANN would not lose generalization ability. The validation set, used for early stopping, was set 2 from table 9. Another training technique is Bayesian regularization which is used to find optimal number of network parameters that are used to model the relationship between independent and dependent variables.

As the number of neurons in the hidden layers of the network has to be decided, ANN was tested with combinations of 1 or 2 hidden layers so that the total product of neurons in the layers would remain between fifth and tenth of the data points used in training. This ensures that the ANN is not overgeneralizing too much as only limited amount of neurons are available. All tests were repeated 5 times with different initial network weights.

Table 11 and 12 presents the results for neural networks. Table shows minimum MSE for each type of data combination and for training and test sets. The number of neurons for the two hidden layers to achieve minimum error are shown in parenthesis.

Table 11: Neural network MSE.

	Test set	Training set
Wear	0.0040 (0,15)	$4.5870 \cdot 10^{-4}$ (0,19)
Wear+Force	0.0029 (1,11)	$2.2373 \cdot 10^{-4}$ (1, 20)
Force	0.0517 (0,10)	0.0027 (0,12)

Tables 11 and 12 show that inclusion of force definitely improves modeling results. With only force measurements, the results are quite poor in comparison to other combinations of data with only early stopping used. However with Bayesian regularization results are almost as good as with other combinations. Bayesian regularization also decreases the error compared to only early stopping.

7.2.4 Support vector regression

Support vector regression was tested similarily to other methods, three combinations of independent variables and the data was scaled to $[0, 1]$. Only Gaussian radial basis function (RBF) as the kernel function was used to test the SVR. Parameters that control the SVR were σ for the Gaussian RBF, ϵ controlling the size of the ϵ-tube and constant C which determines the cost of leaving samples outside of the ϵ-tube. Table 13 shows the best results. The parameters for the results are in parenthesis in order σ, C and ϵ.

The results for SVR in table 13 shows different results than the other two methods. Inclusion of force does not improve the test set results but using force only, the training set error is comparable to other combinations of independent variables.

Table 12: Neural network with Bayesian regularization MSE.

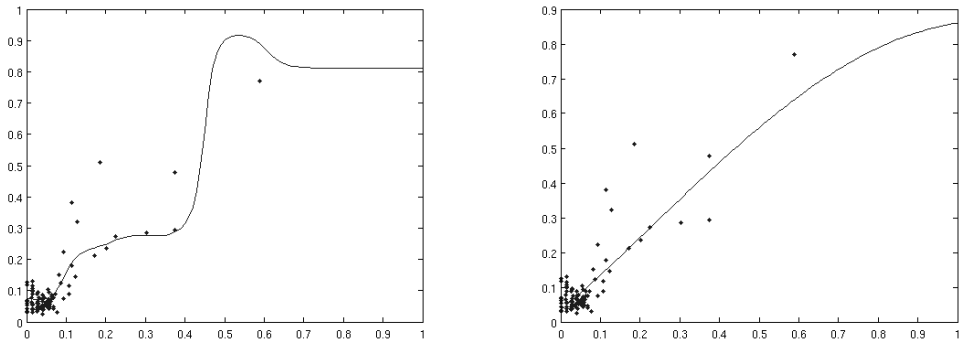
	Test set	Training set
Wear	0.0029 (2,5)	$5.721 \cdot 10^{-4}$ (0,19)
Wear+Force	0.0011 (19,1)	$3.995 \cdot 10^{-4}$ (18,0)
Force	0.0030 (11,1)	$4.799 \cdot 10^{-4}$ (20,1)

Table 13: Support vector regression MSE

	Test set	Training set
Wear	0.0038 (0.9, 15, 0.1)	0.0008 (0.1, 0.5, 0.1)
Wear+Force	0.0038 (0.9, 15, 0.1)	$5.6032 \cdot 10^{-4}$ (0.15, 0.3, 0.1)
Force	0.0133 (0.15, 9.9, 0.1)	$7.6786 \cdot 10^{-4}$ (0.05, 0.3, 0.1)

7.2.5 Comparison of modeling methods as predictors of wear

Sections 7.2.2-7.2.4 present results of the modeling capabilities of three methods. From the MSE tables, it is clear that ANN and SVR are better at modeling the wear speed than linear regression. However, ANN, especially with Bayesian regularization, seems to model the wear speed better than SVR. Figure 53 shows the fit between wear and wear speed created by ANN with Bayesian regularization and SVR with Gaussian RBF. Parameters for both have been selected based on the test set MSE.

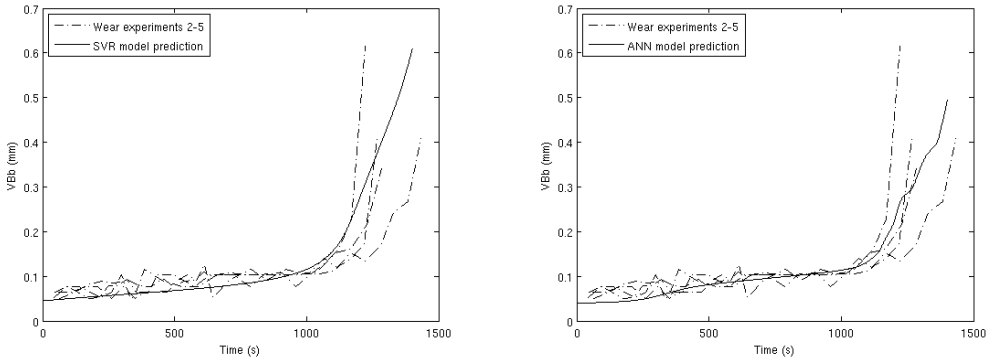


(a) (b)

Figure 53: Model fit: (a) ANN/Bayesian regularization; (b) SVR/Gaussian RBF.

As can be seen from Figure 53, the SVR fit is almost linear, while ANN is more closely following some of the data points. Result of this is that ANN is better at predicting the wear.

The prediction results from both SVR and ANN are quite promising. They are able to model the wear-time-curve quite well as can be seen in Figure 54. Only wear width was used as an independent variable, so force was excluded. However, to get the results shown in Figure 54, starting wear width had to be adjusted. For SVR, the wear width was set to 0.045 mm and for ANN, the wear width was set to 0.041. These values were found manually.



(a) (b)

Figure 54: Prediction by wear models: (a) SVR/Gaussian RBF ; (b) ANN/Bayesian regularization.

Figure 54 shows that the SVR and ANN models of wear speed can translate to a good approximation of the wear-time-curve. Both models exhibit a slight underestimation at the beginning. However, the results are promising for further study with multiple cutting conditions.

8 CONCLUSIONS

The first objective, implementing a system to measure flank wear, was completed successfully. Software that implements the measuring system is easy to extend, or if needed, system components can be replaced. Parameters can be used to adjust the system to different environments. A number of issues, that arose from the hardware components were solved or acknowledged, which will enable further development.

The measurement results that the whole measuring system achieves are satisfactory. Measurements are close to manual measurements which are done with a microscope. However, the measurement system has problems when the flank wear on the tool has not yet formed properly. At current state, the measurement system, depending on cutting conditions, may require a few cuts, before the system can give reliable results. Compared to other similar systems implemented previously, the current system can measure all three, VB_B, VB_B max and VB_C, measurements simultaneously, whereas previous systems only measure one of the measurements.

Although the measurement system was tested only with one type of tool, the system should accommodate any tool that can present a rectangular major flank to the system's camera. With the use of specular reflection, the system should also be able to measure wear on differently colored tools. The measurement system is also capable to withstand the environment of the turning center's work-space, in comparison to the previous systems, which require removing the tool from the turning center for the duration of the measurement.

Modelling experiments showed that ANN and SVR are both viable methods when a differential model is used. Prediction with a differential model proved to be successful as well, with ANN having better performance than SVR. Differential model is different from the majority of previous work on wear modeling, especially work which has been presented in the last few years.

Further study is needed to verify that the prediction is able to predict with several different cutting conditions. After such verification, it is possible to concentrate on the prediction when cutting conditions during turning are changed dynamically, which can be accommodated with the developed differential models.

Another interesting research topic is combining the developed automatic flank wear mea-

surement system with wear prediction and modeling. First of all, the automatic measurement could provide an automatic wear prediction based on the measurement. Secondly, the automatic measurement system can improve the results of prediction. Improving prediction by means of the measurement system can be incorporated with probabilistic methods. These methods can track the uncertainty of the prediction and inform when a new measurement is needed, so that the prediction stays reliable enough.

REFERENCES

[1] International Organization for Standardization. Tool-life testing with single-point turning tools, 1993. ISO 3685.

[2] G. Boothroyd and W. A. Knight. *Fundamentals of Machining and Machine Tools, Third Edition*. Taylor & Francis Group, 2006.

[3] V. P. Astakhov. Effects of the cutting feed, depth of cut, and workpiece(bore) diameter on the tool wear rate. *International Journal of Advanced Manufacturing Technology*, 34:631–640, 2007.

[4] P. Andersson. Lastuamisarvojen ja -häiriöiden korrelaatio. Technical Report 28, Tampere University of Technology, Department of Mechanical engineering, Production Engineering, February 1991.

[5] S. Kurada and C. Bradley. A review of machine vision sensors for tool condition monitoring. *Computers in Industry*, 34(1):55–72, 1997.

[6] A.G. Ulsoy J.J. Park. On-line flank wear estimation using an adaptive observer and computer vision, part 2: Experiment. *ASME Journal of Engineering for Industry*, 115:37–43, 1993.

[7] S. Kurada and C. Bradley. A machine vision system for tool wear assessment. *Tribology International*, 30(4):295–304, 1997.

[8] M. Lanzetta. A new high-resolution vision sensor for tool condition monitoring. *Journal of Materials Processing Technology*, 119(1-3):73–82, 2001.

[9] A.A. Kassim, M.A. Mannan, and Z. Mian. Texture analysis methods for tool condition monitoring. *Image and Vision Computing*, 25(7):1080–1090, 2007.

[10] C. Bradley and Y.S. Wong. Surface texture indicators of tool wear - a machine vision approach. *International Journal of Advanced Manufacturing Technology*, 17(6):435–443, 2001.

[11] A.G. Rehorn, J. Jiang, and P.E. Orban. State-of-the-art methods and results in tool condition monitoring: a review. *International Journal of Advanced Manufacturing Technology*, 26(7-8):693–710, 2005.

[12] K. N. Prasad and B. Ramamoorthy. Tool wear evaluation by stereo vision and prediction by artificial neural network. *Journal of Materials Processing Technology*, 112(1):43–52, 2001.

[13] J. Jurkovic, M. Korosec, and J. Kopac. New approach in tool wear measuring technique using ccd vision system. *International Journal of Machine Tools & Manufacture*, 45(9):1023–1030, 2005.

[14] D. Kerr, J. Pengilley, and R. Garwood. Assessment and visualisation of machine tool wear using computer vision. *International Journal of Advanced Manufacturing Technology*, 28(7-8):781–791, 2006.

[15] S.K. Choudhury and P. Srinivas. Tool wear prediction in turning. *Journal of Materials Processing Technology*, 153-154:276–280, 2004.

[16] J.J. Park and A.G. Ulsoy. On-line flank wear estimation using an adaptive observer and computer vision, part 1: Theory. *ASME Journal of Engineering for Industry*, 115:30–36, 1993.

[17] T. Özel and Y. Karpat. Predictive modeling of surface roughness and tool wear in hard turning using regression and neural networks. *International Journal of Machine Tools & Manufacture*, 45:467–479, 2005.

[18] S.K. Choudhury and G. Bartarya. Role of temperature and surface finish in predicting tool wear using neural network and design of experiments. *International Journal of Machine Tools & Manufacture*, 43:747–753, 2003.

[19] K. Danai and A.G. Ulsoy. A dynamic state model for on-line tool wear estimation in turning. *ASME Journal of Engineering for Industry*, 109:396–399, 1987.

[20] S.E. Oraby and D.R. Hayhurst. Tool life determination based on the measurement of wear and tool force ratio variation. *International Journal of Machine Tools & Manufacture*, 44:1261–1269, 2004.

[21] P. Bhattacharyya and S. K. Sanadhya. Support vector regression based tool wear assessment in face milling. In *Proceedings of the IEEE International Conference on Industrial Technology, ICIT 2006*, pages 2468–2473, 2006.

[22] D. Shi and N. N. Gindy. Tool wear predictive model based on least squares support vector machines. *Mechanical Systems and Signal Processing*, 21:1799–1814, 2007.

[23] Findamachine.com. Daewoo puma 2500y specification. [Accessed: 26.11.2007] Available: http://www.findamachine.com/lathe/DAEWOO/PUMA_2500Y.

[24] R. Jain, R. Kasturi, and B. G. Schunck. *Machine Vision*. McGraw-Hill, Inc., 1995.

[25] D. A. Forsyth and J. Ponce. *Computer Vision*. Pearson Education, Inc., 2003.

[26] J.E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4(1):25–30, 1965.

[27] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[28] R. C. Gonzalez and R. E. Woods. *Digital Image Processing, Second Edition*. Prentice-Hall, Inc., 2002.

[29] A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice-Hall, Inc., 1989.

[30] J. W. Barnes. *Statistical analysis for engineers and scientists: A computer-based approach*. McGRAW-HILL, INC., 1994.

[31] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, Inc., 2001.

[32] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[33] A.J. Smola and B. Schoelkopf. A tutorial on support vector regression, 1998.

[34] IEEE Computer Society. IEEE Std 802.3-2005, 2005.

[35] Intel Corporation. Open source computer vision library. [Accessed: 4.2.2008] Available: http://www.intel.com/technology/computing/opencv/index.htm.

[36] 1394 Trade Association. IIDC 1394-based Digital Camera Specification, 2004. Ver 1.31.

[37] D. van Heesch. Doxygen. [Accessed: 5.2.2008] Available: http://www.stack.nl/ dimitri/doxygen/.

APPENDIX 1. System specification

Specification – Measuring flank wear of a tool inside a turning center

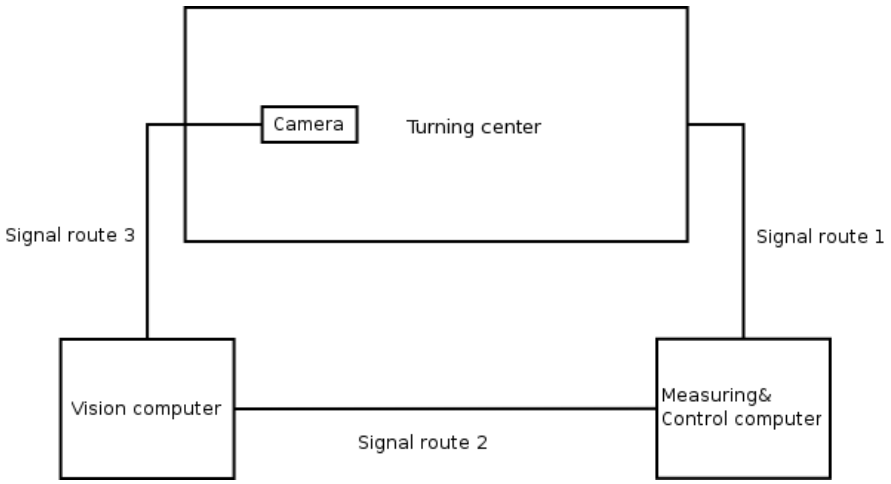


Figure 1: System configuration

Figure 1 shows the intended system configuration. Specifications will be built on this configuration.

Signal specifications:

Route 1:

This is not in the scope of thisproject

Route 2:

Measuring&Control computer -> Vision computer:
Sends message to vision computer that the tool is in the correct position for imaging

Vision computer -> Measuring&Control computer:
Acknowledgement when image has been taken.

Sends wear measurements to measuring&control computer.

Route 3:

Camera -> Vision computer:
Sends image data

Vision computer -> Camera:
Sends settings data

Specifications for individual components:

Vision computer:

FireWire(at least IEEE 1394a) and Ethernet(>=100Mb/s) connectivity
Serial port(RS232)
Linux OS
Processor 2.0 GHz and upwards
Memory 512MB and upwards

Camera and lens:

Firewire connectivity
Color camera
Imaging area: 10mmx7mm
At least 10 um/pixel resolution
Minimum array size(with 10 um pixels, 1.0x): 1000x700 pixels
WD range(from lens): 10-30cm
Fixture to position the camera to correct location and angle relative to the tool
-The camera and lens shouldremain within 25 cm of the turning center's back wall, to protect the camera

Lighting:

Sufficient lighting to light the tool, boundaries should be clear
Specular reflection from worn area is needed
-At least 2 lights required, 1 for rake and 1 for flank face

Functional specification:

Image processing:

Must discern the flank wear zones and measure the extent of wear on each zone(Figure 2)

-Must recognize zones B and C from the image, according to ISO 3685

-Must measure VB_C, VB_B , VB_B max from the image, according to ISO 3685

Should measure the crater wear(distance from crater edge to tool edge)(Figure 3)

-Should measure *KF*, according to ISO 3685

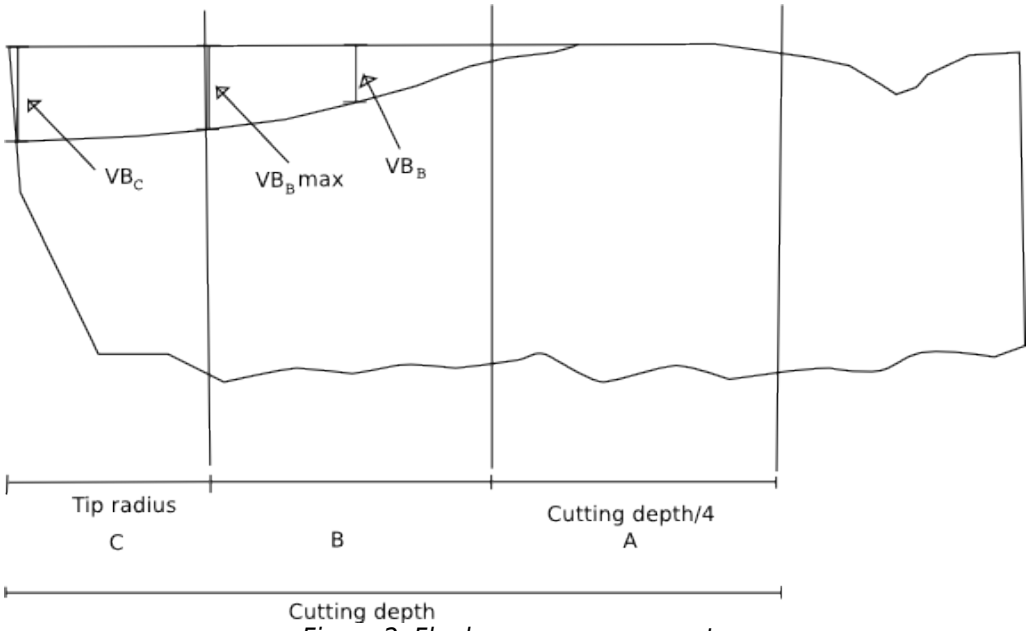


Figure 2: Flank wear measurements

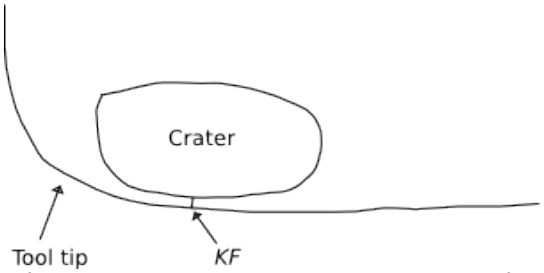


Figure 3: Crater wear measurement on rake face

Communication:

Use TCP/IP between computers to transfer messages to standardize communication.

Must be able to receive messages to initiate imaging of the tool.

Must be able to send messages containing the measurements of the tool and acknowledgment that image has been taken.

Non-functional specification:

Time constraints:

On-line, during production turning:

Image should be processed between the time the tool moves to start a new chip from the time image has been taken. Time estimate 5 seconds.

Off-line, during wear testing:

Image processing should be done before tool is imaged again. Time estimate is upwards from 40 seconds.

User interaction:

System should require user interaction only at the initialization of the software.

APPENDIX 2. Specifications for camera, lens and lighting

Specifications for Moritex ML-Z0108 macro zoom objective

Manufacturer	Moritex
Model	ML-Z0108
Type	0.1 $\times$ - 0.8 $\times$ zoom lens
Working distance	213 mm
WD adjustment	$\pm$ 20 mm
Effective F number	8.2 - 9.3
Depth of field	32.8 - 0.6 mm
Resolution	55 - 8 μm
TV distortion	-0.02 $\pm$ 0.17% (or less)
Largest sensor	1/2"
Mount	C-Mount

Specifications for Allied Vision Technologies GUPPY F-146C Camera

Manufacturer	Allied Vision Technologies
Model	GUPPY F-146C
Type	Industrial camera
Sensor size	1/2"
Resolution	up to 1392×1040
Pixel size	$4.65\ \mu m \times 4.65\ \mu m$
Sensor type	CCD
Colour	RAW
Shutter speed	20 μ s,..., 67 s
Resolution depth	8 bit
Connector	Firewire
Lens mount	C, CS
Dimensions	$48.2\ mm \times 30\ mm \times 30\ mm$

Specifications for Philips Lumileds Luxeon Star LED

Manufacturer	Philips Lumileds
Model	Luxeon I Star
Type	Lambertian
Typical luminous flux	45 lm
Max. DC forward current	350 mA
Operating temperature	-40° to 105°

APPENDIX 3. Algorithms

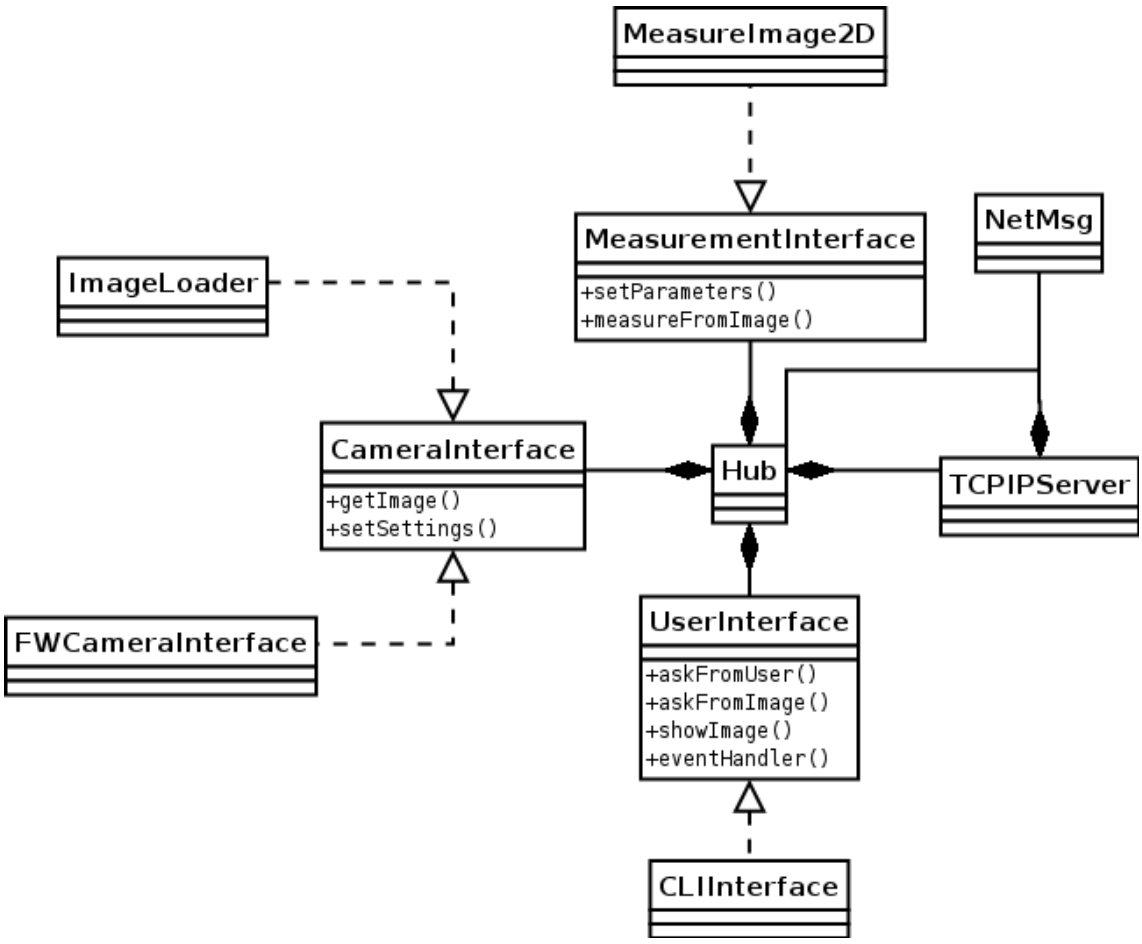
Wear Measurement Algorithm

Algorithm **measure_wear**($img, noseregion, pxmm, pymm$):
Apply $noseregion$ to img
Convert img to grayscale image $gimg$
Adjust $gimg$ brightness and contrast to sharpen edges of the tool
Convert $gimg$ to edge image $eimg$ by using Canny edge detector
Convert $eimg$ to Hough space image $himg$ using Hough transform
Find 2 most prominent lines from $himg$, from both near horizontal and near vertical angles
Using the 2 lines, calculate their intersection, representing the intersection of major and minor flank above the worn area
Extract color channel or convert from img to grayscale image $bimg$
Apply Gaussian smoothing to $bimg$, smoothing out local changes in illumination
Calculate distance representing the known cutting depth a along the tool edge horizontally from the intersection
Solve pixels along the edge,x_i, using Bresenham's line algorithm
for each x_i **do**
    Calculate line from x_i to certain distance d using the angle obtained from the vertical line of the tool, resulting in point x'_i
    Solve pixels between x_i and x'_i, l_j, using Bresenham's line algorithm
    for each pixel l_1 to l_n **do**
        Calculate median from the pixel values (from $bimg$) $l_i - m_s/2$ to $l_i + m_s/2$, m_s median mask size
        if median value changes negatively more than predetermined threshold from the previous round **then**
            End loop, set edge pixel e_i to l_i
        end if
    end for
    if edge pixel e_i has not been set **then**
        set edge pixel e_i to l_n
    end if
end for
Calculate distances from all x_i to their corresponding e_i
Divide the distance along the horizontal edge to regions C and B according to the standard ISO 3685:1993
Use the region and edge distance data to calculate VB_C, VB_B and VB_Bmax according to the standard ISO 3685:1993
return VB_C, VB_B, VB_Bmax

Bresenham's Line Algorithm

```
1:  Algorithm line_(x1, y1, x2, y2):
2:  steep := abs(y2-y1) > abs(x2-x1)
3:  if steep then
4:      swap(x1,y1)
5:      swap(x2,y2)
6:  end if
7:  if x1 > x2 then
8:      swap(x1,x2)
9:      swap(y1,y2)
10: end if
11: deltax := x2 - x1
12: deltay := abs(y2 - y1)
13: error := -deltax / 2
14: y := y1
15: if y1 < y2 then
16:     ystep := 1
17: else
18:     ystep := -1
19: end if
20: for x := x1 to x2 do
21:     if steep then
22:         plot(y,x)
23:     else
24:         plot(x,y)
25:     end if
26:     error := error + deltay
27:     if error ≥ 0 then
28:         y := y + ystep
29:         error := error - deltax
30:     end if
31: end for
```

APPENDIX 4. Software class diagram



APPENDIX 5. Parameter list

Parameter list with default values

Value	Name
300	Camera_Gain
750	Camera_WB1
600	Camera_WB2
3000	Camera_Shutter
4.8	Tool_Width
12.68	Tool_Length
3	Tool_Depth
15	Tool_Angle
1.2	Tool_Radius
96	Pre_Bright
127	Pre_Contrast
300	Canny_Low
300	Canny_High
1	Hough_Vert
89	Hough_Horiz
50	Hough_Search
1	Max_Wear
5	Median_Size
7	Median_Diff
7	Filter_Size
15	Start_Offset
0	Measurement_Show

Parameter list with default values, continues

Value	Name
0	Calibration_Override
300	Calibration_Width
900	Calibration_Length
500	Calibration_ToolX
400	Calibration_ToolY
90	Calibration_Pre_Bright
127	Calibration_Pre_Contrast
300	Calibration_Canny_Low
400	Calibration_Canny_High
11	Calibration_Filter_Size
0	ROI_X
0	ROI_Y
0	Calibration_Show

Parameter descriptions and value ranges

Name	Description	Value range
Camera_Gain	Adjusts camera gain	0-680
Camera_WB1	Adjusts camera white balance	0-1022
Camera_WB2	Adjusts camera white balance	0-1022
Camera_Shutter	Adjusts camera shutter time	1-3000
Tool_Width	Thickness of the tool in mm	-
Tool_Length	Length fo the tool in mm	-
Tool_Depth	Used cutting depth in mm	-
Tool_Angle	Cutting edge angle, K_r, relative to 90°	-
Tool_Radius	Tool nose radius in mm	-
Pre_Bright	Brightness value used in image sharpening	1-127
Pre_Contrast	Contrast value used in image sharpening	1-127
Canny_Low	Low threshold for Canny filter	1-
Canny_High	High threshold for Canny filter	
Hough_Vert	Vertical angle of the tool	0-179
Hough_Horiz	Horizontal angle of the tool	0-179
Hough_Search	Search space for searching tool angles	6 - 60
Max_Wear	Maximum measured wear in mm	0.01 - 2
Median_Size	Size of the 1D median mask	3-15
Median_Diff	Threshold for detecting wear edge	1-30
Filter_Size	2D Gaussian filter size	3-15
Start_Offset	Used for ignoring the unlit wear, in pixels	1-50
Measurement_Show	Shows measurement results in window	0,1
Calibration_Override	Overrides automatic calibration	
Calibration_Width	Thickness of the tool in pixels	-
Calibration_Length	Length of the tool in pixels	-
Calibration_ToolX	Tool nose X position in pixels	-
Calibration_ToolY	Tool nose Y position in pixels	-
Calibration_Pre_Bright	Same as above, but for calibration	-
Calibration_Pre_Contrast	Same as above, but for calibration	-
Calibration_Canny_Low	Same as above, but for calibration	-
Calibration_Canny_High	Same as above, but for calibration	-
Calibration_Filter_Size	Same as above, but for calibration	-
ROI_X	Increases size of the nose region in pixels	1-
ROI_Y	Increases size of the nose region in pixels	1-
Calibration_Show	Shows the calibration result in a window	0,1