Live Tool Condition Monitoring of SiAlON Inserted Tools whilst Milling Nickel-Based Super Alloys

By:
Javier Alejandro Dominguez Caballero

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy.

The University of Sheffield
Department of Mechanical Engineering

September 2017

ABSTRACT

Cutting tools with ceramic inserts are often used in the process of machining many types of super alloys, mainly due to their high strength and thermal resistance. Nevertheless, during the cutting process, the plastic flow wear generated in these inserts enhances and propagates cracks due to high temperature and high mechanical stress. This leads to a very variable failure of the cutting tool. Furthermore, in high-speed rough machining of nickel-based super alloys, such as Inconel 718 and Waspalloy, it is recommended to avoid the use of any type of coolant. This in turn, enables the clear visualization of cutting sparks, which in these machining tasks are quite distinctive.

The present doctoral thesis attempts to set the basis of a potential Tool Condition Monitoring (TCM) system that could use vison-based sensing to calculate the amount of tool wear. This TCM system would work around the research hypothesis that states that a relationship exists between the continuous wear that ceramic SiAlON (solid solutions based on the Si3N4 structure) inserts experience during a high-speed machining process, and the evolution of sparks created during the same process. A successful TCM system such as this could be implemented at an industrial level to aid in providing a live status of the cutting tool's condition, potentially improving the effectiveness of these machining tasks, whilst preventing tool failure and workpiece damage.

During this research, sparks were analyzed through various visual methods in three main experiments. Four studies were developed using the mentioned experiments to support and create a final predictive approach to the TCM system. These studies are described in each thesis chapter and they include a wear assessment of SiAlON ceramics, an analysis of the optimal image acquisition systems and parameters appropriate for this research, a study of the research hypothesis, and finally, an approach to tool wear prediction using Neural Networks (NN). To carry out some of these studies, an overall methodology was structured to perform experiments and to process spark evolution data, as image processing algorithms were built to extract spark area and intensity. Towards the end of this thesis, these spark features were used, along with measured values of tool wear, namely notch, flank and crater wear, to build a Neural Network for tool wear prediction.

ACKNOWLEDGMENTS

Firstly, I would like to thank Dr Graeme Manson and Dr Matthew Marshall, who's support, encouragement and guidance led me throughout the duration of this study.

Secondly, I would like to thank Jamie Booth for his patience and support during the long experimental work at the workshop.

I would also like to thank CONACYT and Roberto Rocca Fellowship, for the financial support during this degree, as well as FRISA for the support in providing the required materials.

A special thanks to my friends and colleagues Mariana, Antonio, Miguel, Elda, Fernando, Elizabeth, Carlos, Omar and Daniel for their great support, and most importantly, for their friendship.

My most deep gratefulness to my family, who's support during all my life has always been strong, ever present and immeasurable. My parents, who have always believed in me and my dreams, providing the best childhood I could ever have and an amazing example of humility, compassion, intelligence and true love. And my brothers, Susana, Pepe y Carlos, who's invaluable support and love has truly guided my life.

Finally, my most sincere and grateful thanks to my wife Zusej, for sharing this adventure of professional development with the upmost support and patience. Thank you for your encouragement, your never-ending tolerance, and most importantly, your incredible love, which inspires me and fills our small family with happiness.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

$\propto$	Netlab weight-decay regularisation hyperparameter
AE	Acoustic Emissions
ae	Radial Immersion (mm)
$a_j^{(\mathrm{L})}$	NN activation variables for layer L
AMRC	Advanced Manufacturing Research Centre
ANFIS	Adaptive Network-Based Fuzzy Interference Systems
ANN	Artificial Neural Network
ANOVA	Analysis of Variance
ap	Machining cutting Depth (mm)
B	Baseline, distance between the two cameras
bw	Binary image
cc	Resulting image from connected components filtering
CCD	Charge-Coupled Device
CNC	Computer Numerical Control
COF	Coefficient of Friction
C_t	Taylor's constant for tool life
D	Diameter of cutting tool (mm)
DIP	Digital Image Processing
$\delta_k^{(\mathrm{L})n}$	NN difference between output and target values
D_0	Distance to lowest frequency value
DOC	Machining radial depth of cut (%)
DOE	Design of Experiments
E	Netlab overall error or Bayesian error

E_D	Netlab data error
EDX	Energy Dispersive X-Ray
E_W	Netlab weight error
f	Focal length of a camera
fz	Machining feed per tooth (mm)
HCRR	High Chip Removal Rate
H_{hp}	High-pass frequency filter
H_{lp}	Low-pass frequency filter
HRSA	Heat Resistant Super Alloys
I	Digital image
I_g	Grey-scale image
I_{RGB}	RGB image
I_{subt}	Resulting image from image subtraction
KB	Distance from the cutting edge to the back crater contour
KE	Radial displacement of the tool
KF	Width of the land between the crater and cutting edge
KM	Distance from the cutting edge to the deepest crater point
KT	Crater depth
MLP	Multi-Layer Perceptron
MQCL	Minimum Quantity Cooling Lubrication
MSE	Mean-squared error
n	Machining spindle speed (rpm)
NN	Neural Network
P	Real-world point defined by the coordinates x_R, y_R, and z_R

R	Reference image
RVM	Relevance Vector Machine
SEM	Scanning Electron Microscope
SLR	Single Lens Reflex
SVM	Support Vector Machine
T	Tool life expectancy (s)
t	NN target values
TCM	Tool Condition Monitoring
$threshold$	Threshold of intensity discrimination for binarization
U_L	Coordinate of the real-world point P in an image acquired by the left camera
U_R	Coordinate of the real-world point P in an image acquired by the right camera
VB_B	Average flank wear land width
VB_{Bmax}	Maximum flank wear land width
VB_C	Width of flank wear at tool corner
VBN	Width of notch wear
V_c	Machining cutting speed (m/min)
Vf	Machining feed per minute (mm/min)
w	NN weight values
x_i, x_0	NN input layer i values and input layer bias
$X_L,\ X_R$	X-axis of Left and Right camera
y_k	NN output layer k values
zc	Number of inserts in cutting tool
z_j, z_0	NN hidden nodes layer j values and hidden nodes layer bias
Z_L, Z_R	Optical axis of Left and Right camera

1 INTRODUCTION

1.1 Study Motivation

Today's manufacturing industry continues to have a strong dependence on traditional machining processes for piece formation such as Turning and Milling, and when great productivity is demanded, a High Chip Removal Rate (HCRR) becomes essential. Researchers have hence explored the effect of different machining properties and factors to achieve HCRR, as illustrated by Figure 1.1 (Grzesik 2008; Dashchenko 2012). There are several different parameters associated with the machining process, such as feed, depth of cut and cutting speed, which could be altered to obtain a high chip removal rate. The most common choice for increasing removal rate is to increase the cutting speed, given that increasing the other two parameters could result in a wider chip cross-section, and hence much higher cutting forces (Liu et al. 2002; Casto et al. 1993; Altin et al. 2007). High-speed machining, which produces high chip removal rates with low energy consumption, has proved to deliver these outcomes (X. Tian et al. 2013; Addhoum & Broussaud 1989).

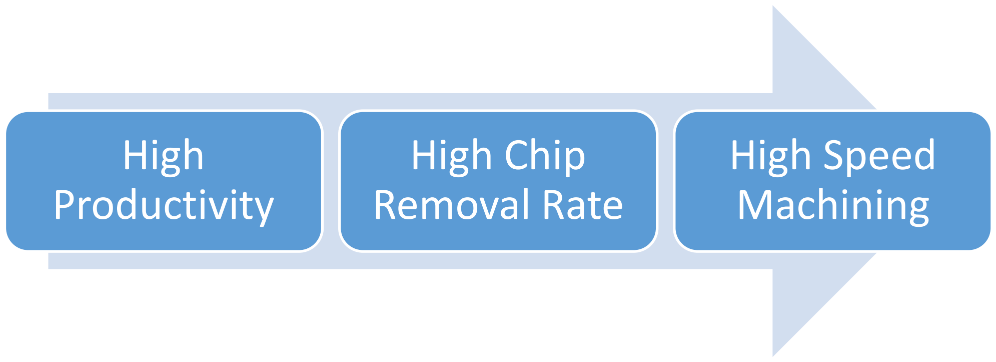


Figure 1.1: Machining sequence of factors for high productivity.

However, when the cutting speed is increased, this results in higher stresses and higher temperatures in the cutting area, therefore demanding greater strength and thermal resistance from the cutting tool. Traditional cemented tungsten carbides have good performance up to around 800 °C however, at higher temperatures, their strength decreases dramatically. By contrast, ceramic cutting tools show good performance up to 1200 °C (Casto et al. 1993). These tools are hard materials with high hot-hardness that are unreactive with the workpiece. More importantly, these materials can machine at high cutting speeds, as they can work at 20 to 30 times faster than carbides (Sandvik Coromant n.d.; Sandvik Coromant 1994). Even though ceramic tools have a short tool life regarding machining time of between 4 to 8 minutes, they more than compensate this when compared to any other carbide tool, making them cost-effective (Richards & Aspinwall 1989). In the case of *heat* *resistant super alloys (HRSA),* in the current aero-engine manufacturing industry, the performance temperature for efficient combustion regimes has increased. This has resulted in tougher HRSA that are harder to cut and hence, slowing machining removal rates. Ceramics, however, have proved to enable very high-speeds, balancing performance and price (Bitterlich et al. 2008; Renz et al. 2015).

SiAlON ceramic cutting tools are solid solutions based on the Si3N4 structure and are widely used in high-speed milling of nickel-based super alloys. This material has a high fracture toughness, hot strength and great thermal shock resistance, making them very efficient in rough milling alloys such as Inconel 718 and Waspalloy (Zheng et al. 2012; Xianhua Tian et al. 2013; Arunachalam & Mannan 2000). However, the plastic flow wear, due to high temperature and high mechanical stress generated in these cutting tools, enhances and propagates cracks. This condition can sometimes result in unpredictable wear, which could, in turn, result in catastrophic failure and possible damage to the workpiece.

Kurada and Bradley (1997a) state that “Timely change of the cutting tool toward the end of its useful life prevents inferior surface finish quality (leading to scrap and re-work costs) or worse, damage to the machine tool itself.” Hence, tool replacement can be of extreme importance in machining processes. Furthermore, tool replacement can be catalogued into two categories, depending on the value of the workpiece. In the first, the workpiece value may be lower than the cost of machine downtime and tool change. This scenario is commonly found in a large production environment, where a constant flow of a manufacturing chain is highly important. Therefore, in these cases, a complete degradation of the tool is generally desired, regardless of potential damage to a workpiece. On the other hand, in the second category, the value of the workpiece is comparatively higher than possible machine downtime. In these cases, possible damage to the workpiece could be costly, and hence the tool has to be changed before unacceptable results are reached (D’Addona & Teti 2013). Workpieces composed from HRSA are regarded in the second category, as the value of these materials is extremely high, compared to the cost of a cutting tool such as SiAlON ceramics. For this reason, *Tool Condition Monitoring (TCM)* has become an important area of study in the manufacturing industry (Byrne et al. 1995). By continuously monitoring the actual wear that cutting tools develop, it would be possible to prevent either discarding of the tools when they are still usable or, alternatively, over-usage that could lead to failure and possible damage to the workpiece. This last is especially important with ceramics’ cutting speeds, where if a tool fails at a cutting speed of around 1000 m/min, the possibility of creating significant damage to the workpiece and scrap it is quite high, compared to slower speeds with carbides where it is less severe and unlikely to be scrapped.

Over the years, there has been a continuous transformation of manufacturing machines, evolving into more automated and unmanned systems. In modern industry, 20% of the downtime in production is attributed to tool failure and TCM systems could help prevent this by being integrated with an automation of manufacturing machines (Kurada & Bradley 1997b). Furthermore, TCM can be divided into two areas: direct and indirect monitoring. In Direct methods, tool wear is measured by direct assessment of the worn cutting tool, using techniques such as optical measurements in a microscope or tactile sensors. Indirect methods, on the other hand, try to use parameters or signals such as acoustic emissions, cutting forces and visual sensors (acquiring visual data of other factors other than the actual cutting tool) to evaluate the state of the worn tool (Pfeifer & Wiegers 2000); where this last indirect technique of visual sensing has become the centre of study of the present research.

Using *Digital Image Processing (DIP)*, visual sensors can perform a live and continuous extraction of information from areas of interest, such as a cutting tool in a machining process (Kurada & Bradley 1997a). DIP refers to the processing of digital images through a computer, generally done through software that can manipulate the structure of the images and extract relevant information (Gonzalez & Woods 2010). The relatively low cost and high availability of vision sensor devices, such as CCD cameras or SLR cameras, has enabled their use in different aspects of TCM. At present, however, most of the vision-based TCM rely on processing direct images of the cutting tools; and to obtain these images, the machining process has to be interrupted and the tool removed (D'Addona & Teti 2013; Kurada & Bradley 1997b; Pfeifer & Wiegers 2000; Teshima et al. 1993). This can be a time-consuming process that can reduce cost-effectiveness in a process. As a representative example with information provided by Flores (2015), technical staff in the Mexican metal forging company FRISA use ceramic inserts during a standardised period of 5 minutes to prevent tool failure, regardless of the actual ceramic tools' condition. Furthermore, tools are inspected every minute, with an average inspection time of 30 seconds and a mean insert changing time of 5 minutes. Therefore, considering an average of 200 inserts used per day and a machining cost of 50 USD/hour, a successful low-cost live monitoring system could significantly improve machining cost effectiveness. This is apart from possible damage to workpieces that normally have a value of thousands of dollars. It is therefore also relevant to mention that the wear of ceramic tools is not fixed with time, as it varies greatly depending on machining parameters and the cutting tool composition.

The milling of nickel-based super alloys with SiAlON cutting tools is generally performed without the use of coolant, which could generate a thermal shock that could enable tool failure. This dry milling condition, however, enables the formation of large and very visible sparks emanating from the cutting area as shown in Figure 1.2.

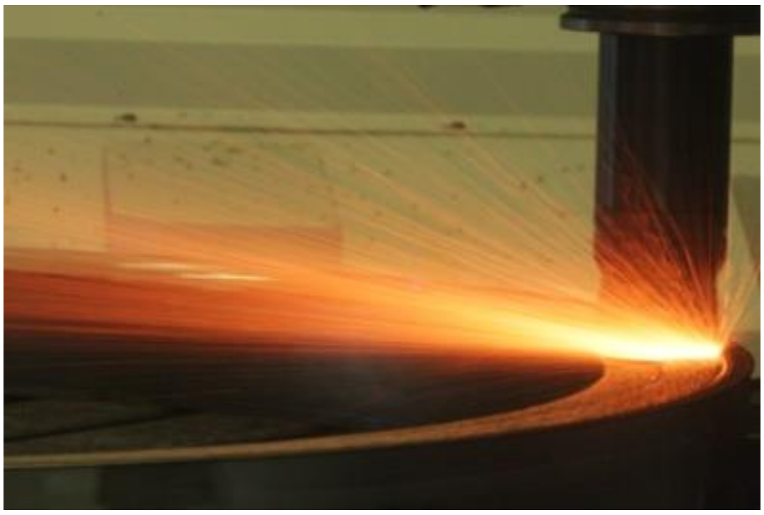


Figure 1.2: High-speed dry milling process of nickel-based super alloy using SiAlON cutting tools.

Therefore, the present doctoral thesis intends to examine the main research hypothesis of *an existing intrinsic relationship between the SiAlON ceramic cutting tool wear and the behaviour of the milling process' cutting sparks*. Through several machining experiments, different research objectives were analysed to support this hypothesis and develop the basis of a vision-based TCM system. Therefore, initially, a tool wear assessment was performed to understand and categorise tool wear mechanisms, presenting as well relevant methods of tool wear measurement. Then the behaviour of cutting sparks through computer vision and image processing was examined, and the suitable image acquisition systems and parameters were assessed. This spark behaviour would be later compared to the measured tool wear to understand the relationship established by the research hypothesis. Finally, a Multi-Layer Neural Network was developed and implemented to predict cutting tool life. The ultimate objective of this research would be to create a fully integrated live TCM that could be implemented at an industrial level. Such a system could be used by technical staff as a companion instrument that could increase machining time and prevent tool failure. Alternatively, machining sequences could be automatically modified to include a tool change in response to a wear level signal provided by such a TCM system, given the capabilities of the modern machines of online integration with other systems.

1.2 Aim and Objectives

Given the previously stated research hypothesis, the main aim of this work was:

To set the basis for a potential live tool condition monitoring system that, using digital image processing to analyse the cutting sparks behaviour, could predict the tool life of SiAlON ceramic inserts whilst high-speed milling nickel-based super alloys.

Where the main research objectives were:

1. To review existing literature on TCM systems, specifically research involving vision-based TCM systems.
2. To plan and carry out experimental testing to collect cutting spark data and tool wear data.
3. To assess the wear of SiAlON ceramic inserted tools whilst analysing the appropriate wear measurement techniques for data collection.
4. To evaluate and select the optimal image acquisition system and parameters, appropriate for the desired application.
5. To evaluate and give evidence of the existence of a spark – tool wear relationship to support the research hypothesis.
6. To implement a machine learning algorithm for tool life prediction and evaluate its accuracy.

1.3 Thesis Structure

As per the research aim and objectives, this section will summarise the structure that was chosen for the present written work. It is important to mention that, after careful and extended consideration, the traditional chronological layout of a doctoral thesis was not selected. During the four years of the doctoral programme, tackling each research objective mentioned above was highly dependent on the decisions and planning behind experimentation, material procurement and data processing. The main issue was that of material procurement, which took unexpected periods of time, and therefore, the order of the research activities had to be adjusted. For this reason, the present thesis was divided into a series of semi-independent studies, each using data from the different experiments carried out. This structure was deemed to be clearer for the reader, trying to avoid repetition and creating a web of support and justification between chapters.

Chapter 2 attempts to give a broad picture of the literature that surrounds and supports this work. It includes a selection of articles about ceramic cutting tool wear and tool wear measurement, as well as literature on tool condition monitoring, including articles of vision-based, direct and indirect methods of TCM. Towards the end of that chapter, a review of the theoretical background of digital image processing and neural networks is included.

Chapter 3 includes the methodology followed during experimentation and data processing, where a general experimental set-up is presented and explained, the data processing methods are stated, and the specific resources and parameters of each experiment are mentioned.

In chapter 4, two techniques for measuring tool wear are explored, one using image processing to perform geometrical measurements, and the other technique includes an attempt to calculate wear volume through a stereovision approach. The successful techniques for wear measurement were used in subsequent chapters to extract wear data from experimental testing. Additionally, this chapter includes a wear assessment of SiAlON ceramic cutting tools, identifying the different primary and secondary wear mechanisms involved. The results obtained were compared and contrasted to the literature found. The findings that this chapter delivered were of high relevance for the final chapter of this thesis, supporting some of the decisions made.

Chapter 5 carries out a study of the different image acquisition systems used in experimentation, analysing their strengths and limitations, as well as examining the parameters used in those systems. The chapter is subdivided into two analyses, corresponding to two experimental datasets, with the objective of selecting the optimal systems and parameters for the present research. The results and conclusions found in this chapter were of importance for the planning of other experimental tests, and the optimisation of the image processing algorithms used.

Chapter 6 is also subdivided into two analyses with the objective of exploring and supporting the main research hypothesis. Two experimental datasets are used, processed and analysed to give evidence of the spark – tool wear relationship. This is a crucial chapter, as it would set the basis of the entire concept of a vision-based TCM system, cementing the foundations of the subsequent and final chapter.

Chapter 7 combines the conclusions found in the results of previous chapters, as a Neural Network (NN) was implemented to predict the tool wear of ceramic inserts. At the beginning of this chapter, some testing parameters and a revision to the methodology used in previous tests was described. Then, the fundamentals of two-layer neural networks were included, along with the structure of the implemented NN and a general NN algorithm. After this, the NN implementation and their results were divided into three approaches to understand the network's behaviour. The first approach created individual networks for each test, whilst the second approach used randomised data points of the all the tests to build a single NN. The third and final approach included the full information of individual tests to build a single NN. All these approaches included different results of accuracy, and therefore, some important conclusions will be found in this chapter.

The final chapter 8 of general conclusions include a summary of all the discussions written in all the chapters of this thesis, followed by three sections: contributions, further research implications and further implementation implications. The second section refers to the future work and implications found if the research was continued research, whilst the third section refers to the implications of the implementation of the proposed system in a more realistic and industrial scenario.

1.4 Background

The research work presented in this thesis is a continuity of the authors MSc dissertation project. That research programme included a partnership with the University of Sheffield's Advanced Manufacturing Research Centre, as the research project was in conjunction with them. It was there that the possibility of a spark - tool wear relationship was identified, as some technical staff could identify the wear condition of cutting tools with good accuracy by simple visual inspection. This motivated the idea of searching the features and relations that their expert eyes identified and try to replicate this for an automated system of tool condition monitoring.

Inside that study, an initial approach to the basis of this tool condition monitoring system with image processing of sparks was established. Through the set of experiments described in section 3.3.1, it was possible to create the initial version of the image processing algorithms that would be optimised in this work. However, at that moment, only the spark intensity approach was explored, performing an analysis of the appropriate filtering techniques. Furthermore, the initial foundations of the crater wear area measurement approach described in section 4.1.1 were established.

It was found in that initial study that the spark intensity was an appropriate feature for successful qualitative correlation with tool wear. Inside the mentioned study of the optimal filtering techniques, three frequency domain filters were tested, namely a high-pass filter, band-pass filter and low-pass filter. It was concluded that the best filtering technique for spark intensity extraction was a low-pass filter, which will be furtherly described in section 2.3.2 and implemented in section 3.2.2. Additionally, given that an initial comparison of spark evolution and tool wear was carried out, it was found that the main research hypothesis could be supported, but that more work was needed. This resulted in the continuation of this research into the present doctoral study.

2 LITERATURE REVIEW

This chapter provides a critical review of the relevant literature along with justification for some of the choices made by the author during the experimental work. However, whilst the bulk of the relevant literature is discussed in this chapter, there will be a discussion of other relevant literature at upcoming chapters.

2.1 Ceramic Cutting Tools and Wear

As proposed in chapter 1 and described in section 1.4, the present research includes different aspects of tool wear condition monitoring. For this reason, it is important to lay out the foundations behind tool wear (with special regard to ceramic cutting tools and nickel-based super alloys), as well as general tool wear assessment theory with an interest in SiAlON ceramics. The focus on wear in SiAlON ceramics will be of great relevance for the assessment and discussion conducted in chapter 4.

With the constant development and evolution of advanced manufacturing technologies, in conjunction with the pursuit of high productivity, the reduction of manufacturing costs and energy consumption have become widely desired. In metal cutting, this has translated into advanced machining technologies, where chip removal rates can be increased without sacrificing workpiece accuracy (Addhoum & Broussaud 1989; X. Tian et al. 2013a). There are two main methods of achieving high chip removal rates. These are (1) increase cutting speeds or (2) increase chip cross section. The second method is less desirable due to deflection and stability constraints, whereas the increase of cutting speeds has been the preferred option for modern industry, translated into what is known as high-speed machining. However, the increase of cutting speeds creates high stresses and high temperatures in the tool-workpiece interface, resulting mostly in tool material softening (Addhoum & Broussaud 1989; Liu et al. 2002).

High-speed machining has been applied in many manufacturing industries. One of these is the aerospace industry, where Heat Resistant Super Alloys (HRSA) are commonly used for many components (Liu et al. 2002). Nickel-based super alloys are a type of HRSA widely used for engine parts due to their high-temperature strength, high toughness and resistance to degradation by corrosion or oxidation (Zhu et al. 2013). Therefore, any machining process of these super alloys demands cutting tool materials that can cope with these workpiece properties whilst withstanding the extreme process conditions mentioned above.

2.1.1 Ceramic Cutting Tools: SiAlON

Ceramic cutting tools have been found to possess excellent material properties for the turning and milling of super alloys and other hard materials. For instance, ceramic tools soften at about2200 °C, whereas carbide tools soften at about 870 °C. This condition is known as hot hardness, and Figure 2.1 shows the Vickers hardness against temperature for different cutting tool materials (Grzesik 2008). Ceramic cutting tools also demonstrate a good balance between performance and cost, being able to bear cutting speeds 20 to 30 times faster than carbides, making them quite cost effective (Casto *et al*>, 1993; Sandvik Coromant, 2010a).

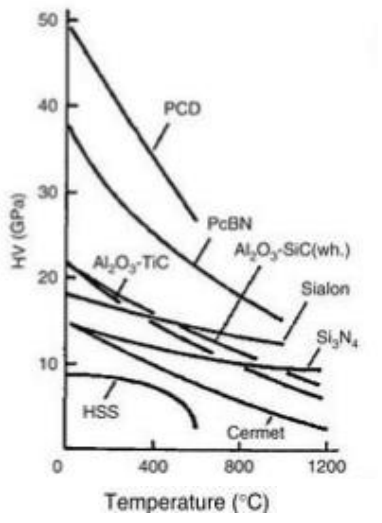


Figure 2.1: Tool hardness (Vickers) against temperature (°C) for different cutting tool materials (Grzesik 2008)**.**

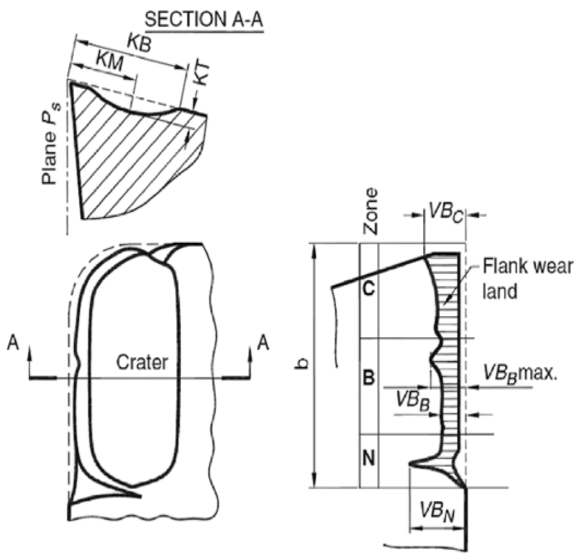
Ceramic cutting tools can be divided into two main groups: ceramics based on aluminium oxide (Al_2O_3), and the ones based on silicon nitride (Si_3N_4). This last group has high fracture toughness, high strength and very good thermal shock resistance, making them able to maintain their hardness at very high temperatures (Grzesik 2008). SiAlONs are solid solutions based on the silicon nitride and are broadly used for machining nickel based super alloys. They are more chemically stable than fully dense silicon nitride tools, with high strength and fracture toughness (Grzesik 2008; Zheng et al. 2012). However, SiAlON cutting tools, as well as most other ceramic cutting tools, tend to have a short tool life. Richards & Aspinwall (1989) compared a machining process of a turbine engine compressor disc using carbide tools and SiAlON tools. Their results showed that carbide tools gave a tool life of around 20 min whilst the ceramic cutters only gave 4 min of time life. However, when the machining parameters of speed and feed were compared, there was an increase by a factor of seven. In recent years, these cutting tools are extensively used in rough milling due to their cost effectiveness, being faster than regular carbides (Arunachalam & Mannan 2000).

2.1.2 Cutting Tool Wear

The cutting parameters selected for every high-speed machining process are generally maximised to assure high productivity. This condition creates extreme cutting conditions where tool life becomes shortened. These conditions include high temperatures, prone to oxidation, diffusion and thermal wear and high sliding contact pressures that promote intense tool wear, leading to tool failure. Therefore, tool wear becomes a rich tribological phenomenon where different wear mechanisms can generally be identified (Astakhov 2006).

Regarding tool wear, there are six different *secondary* tool wear mechanisms shown in Figure 2.2, which are standardised by ISO 3685 and are widely studied and used to determine tool properties and tool life:

1. Crater wear (or rake face wear) is found on the rake face of the cutting tool, generally caused by the chemical interaction of the insert's rake face and the hot chip.
2. Flank wear is found on the clearance side of the cutting tool and it is promoted by abrasion and sliding wear between tool and workpiece. This mechanism is commonly used in wear assessment studies of cutting tool materials.
3. Built-up edge occurs when machining low carbon steels and nonferrous materials at low speeds, and where there is welding of the material chip onto the cutting tool.
4. Notch wear can be found at the side of the flank face of the cutting tool, and it is mainly due to the abrasion of the hard, outer edge of the hot chip. This is especially serious with nickel-based super alloys, as they have high work-hardening properties and generate high cutting temperatures. This mechanism can lead to tool failure due to tool fracture and can be minimized using round inserted tool.
5. Nose wear tends to occur in pointed cutting tools, where there is a deformation of the cutting tool edge.
6. Thermal cracks are mainly present in cyclic processes like milling, where there is a thermomechanical fatigue from the constant heating and cooling of the tool. This mechanism can lead to the dilapidation of the cutting tool due to the growth of these cracks (Grzesik 2008).



Geometrical indicators of tool wear:
VB_B – Average flank wear land width.
VB_{Bmax} – Maximum flank wear land width.
VB_N – Width of notch wear.
VB_C – Width of flank wear at tool corner.
KT – Crater depth.
KM – Distance from the cutting edge to the deepest crater point.
KB – Distance from the cutting edge to the back crater contour.
KF – Width of the land between the crater and cutting edge.
KE – Radial displacement of the tool corner.

Figure 2.2: Types of tool wear per ISO 3685.

Flank wear is the tool wear mechanism mostly used for assessment, as it can be progressive and easier to monitor. In Figure 2.2 it can be seen how this secondary wear mechanism is divided into 3 zones: zone C is identified in this case for squared or pointed tools and this zone is where the rounded part of the cutting edge or corner is found, zone N includes the notch wear, and zone B is where assessment measurements are extracted. VB_B is the average flank wear, broadly used to evaluate tool wear development and tool life, nevertheless, VB_{max} can also be used as tool life criterion by means of a threshold.

Furthermore, the primary wear mechanisms that are present in these secondary tool wear mechanisms can also be classified into five main groups:

- Abrasive wear from the relative motion between tool and workpiece, promoting material loss from the sliding of hard particles of the workpiece with the cutting tool; mostly present in flank wear.
- Adhesive wear can be found as part of the built-up edge secondary tool wear mechanism, where fragments of the hot chip adhere to the cutting tool.
- Diffusion wear is a thermally-activated mechanism where atom migration occurs, however, this primary mechanism is more commonly found in carbide and tungsten carbide tools.
- Oxidation wear is also promoted by high temperatures, it depends greatly on tool-workpiece materials and cutting conditions. It consists in a chemical reaction between the cutting tool and workpiece materials, along with exposed air.
- Thermal wear that occurs due to thermal softening, leading to plastic deformation or material fracture. (Altintas 2000; Grzesik 2008)

All these primary wear mechanisms and their relation with cutting temperature can be represented by Figure 2.3, which is a generic figure by Grzesik (2008) for cutting tools in general. Therefore, this figure would vary depending of tool wear materials. However, a similar graphic could not be found specifically for ceramic tools.

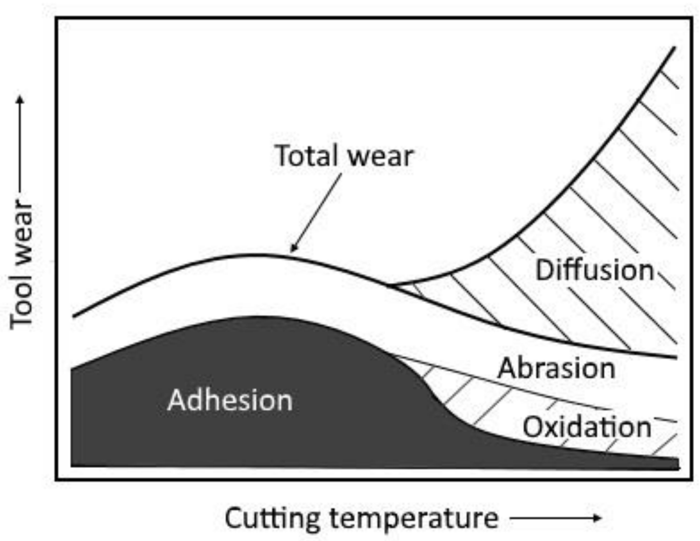


Figure 2.3: Machining physical primary wear mechanisms as a function of cutting temperature (Grzesik 2008)**.**

Authors like Casto et al. (1993) and Zhu et al. (2013) have investigated the characterisation and study of ceramic tool wear mechanisms and general tool wear in machining nickel-based alloys. However, more detailed and earlier assessments of SiAlON's can be found in the work of Bhattacharyya et al. (1983) and Aucote & Foster (1986) where both deepened into the tool material composition and its wear mechanisms, finding a dominating presence of diffusion wear, abrasion, plastic deformation and chemical wear. However, a more updated and detailed review of recent studies of SiAlON cutting tool wear will be presented in the next section.

Finally, it is important to mention that a crucial objective of tool wear assessment is the determination of tool life. While this will not be taken as part of the scope of this research, it is important to lay out the fundamentals of this for future work. Tool life defines the corresponding phases of a tool's wear in a continuous manner (through time). Figure 2.4 shows tool life curves with the different phases or wear zones that a tool may display, taking the value of flank wear with respect to cutting time. It can be seen how three main zones include a primary wear zone, better known as the running-in period of the cutting process. Then there is the secondary wear zone, where the flank wear is more uniform and progressive; whereas in the tertiary zone, the flank wear increases dramatically, generally leading to catastrophic failure.

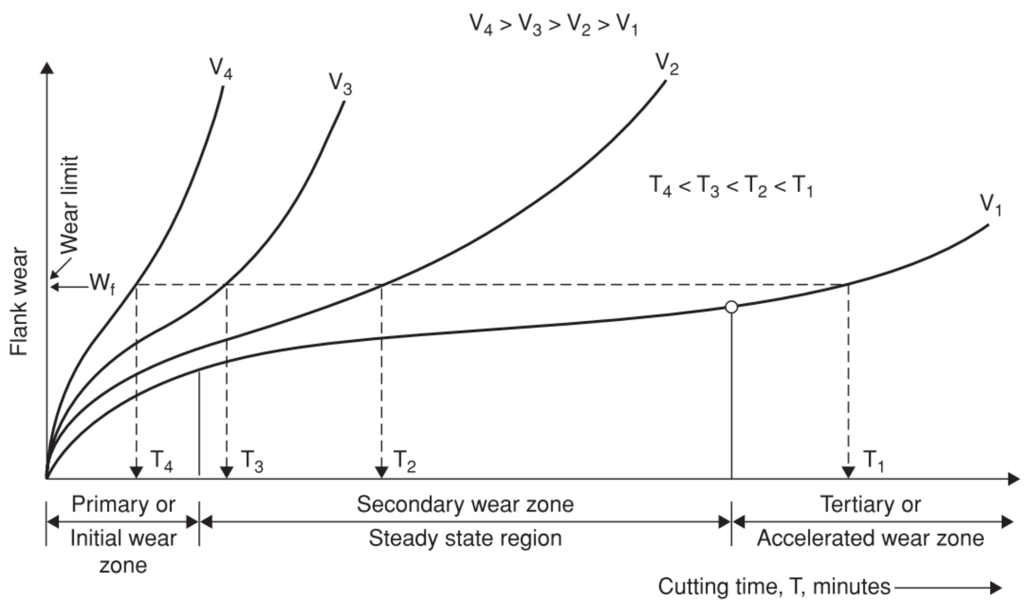


Figure 2.4: Tool life curves (Ti) at different cutting speeds (Vi) (Altintas 2000)**.**

The most commonly used tool life model is given by Taylor's tool-life (Equation 2.1), where a tool's life (T) is as expected, inversely related to the cutting speed of the process (V_c).

$$V_c T^n = C_t$$

Equation 2.1

Where C_t is a constant known as Taylor's constant that represents cutting speed for one-minute tool life, and n is an exponent depending on the material tested.

2.1.3 Wear of SiAlON Ceramic Cutting Tools

The different primary and secondary wear mechanisms of SiAlON ceramic tools cutting Inconel 718 found in literature will be discussed fully in the chapter 4. However, it is appropriate to include a summary of these articles in the present section.

Three pieces of research were found to be of relevance in the present research, as they include the exact same cutting tools and workpiece materials. Tian et al. (2013b) tested various cutting speeds, from 600 m/min to 3000 m/min, analysing as well cutting forces and finding secondary wear mechanisms such as crater wear, flank wear, notch wear and microcracks. These secondary wear mechanisms would vary between cutting speeds. Zheng et al. (2016) also tested various cutting speeds, but in a lower range from 200 m/min to 1000 m/min, and found the same secondary wear mechanisms also dependant on the different cutting speeds. Finally, Renz et al. (2015) carried out a tribochemical approach and found different phases of the tool's wear at different cutting speeds, where evidence of lubricous tribolayers was found. Again, these findings will be discussed and explained furtherly in the chapter 4, where they will be compared to the present research's findings.

2.2 Tool Condition Monitoring

Referring again to the aims and objectives of the present research, this work consists in developing the foundations of a Tool Condition Monitoring (TCM) for the assessment of tool wear. Having discussed the literature on tool wear, this section will now investigate the area of Tool Condition Monitoring (TCM). This section centres on exposing aspects of TCM and its application to machining operations, with a final section on the research that relates more closely to this work.

The constant evolution and development of different advanced sensing devices have made the task of condition monitoring of different processes and tasks more diverse and accessible. Also, the different data and signal processing algorithms that can be implemented in software programmes, have opened new possible ways of monitoring. In the specific area of machining, the introduction of high-speed machining has made tool condition monitoring very important (Byrne et al. 1995).

According to Ambhore et al. (2015), the different methods that can be used to monitor tool condition can be sorted into direct and indirect methods. Direct methods involve direct contact with the tool and process, requiring direct measurements and manipulation of the cutting tools. These methods include sensing techniques such as electric resistance, optical sensing, radioactive sensing, tool geometry measurements and vision systems. However, these methods deal with the ever-present difficulty of lack of access to the actual cutting area of a machining process, nevertheless they tend to be more accurate. Indirect methods like cutting forces, vibrations, temperature analysis, acoustic emissions, surface roughness and vision systems, on the other hand, are not directly measured and signal processing techniques are needed to analyse and correlate these signals to actual conditions. Vision systems appear in both lists because they can be used to directly photograph cutting tools, in the case of direct methods, or else record external features of the machining process that can be related to tool wear, such as surface roughness, or in the case of this research, cutting sparks evolution. Ambhore et al. also mentioned that these methods typically follow the general sequence in systems of tool condition monitoring shown in Figure 2.5. It is however in the last four sections where most of the research in TCM tend to concentrate.

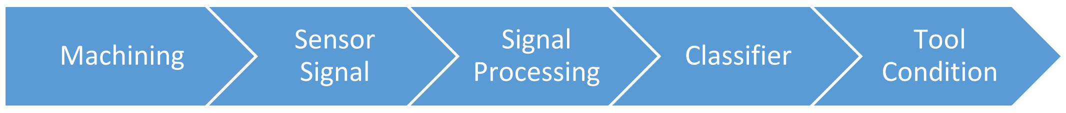


Figure 2.5: Tool condition monitoring systems general sequence (Ambhore et al. 2015)**.**

The search of an accurate and efficient TCM system is the main objective of most authors. Nevertheless, due to the great variety of machining processes, as well as the different tool materials, their geometries and the workpiece materials, most proposed systems seem to only be applicable to the certain conditions presented by their authors.

Direct methods of measuring tool geometry are perhaps the most widely used monitoring and inspection methods. As it was reviewed in section 2.1.2, there are some secondary wear mechanisms that are generally used for wear assessment and monitoring, such as flank wear (either average or maximum) and crater wear (area or depth). Therefore, most of the direct methods consist in directly measuring, in some manner, the geometry of one or all of these secondary wear mechanisms. In the next section, there will be some examples of these methods, using vision systems and algorithms; however, since the main research of this thesis concentrates in the creation of an indirect TCM method, there will be a deeper analysis of this second group.

Inside the indirect TCM systems, the most commonly used and researched are cutting forces, vibrations and acoustic emissions. For instance, Ko & Koren (1989), in a very early publication of TCM, compared and correlated the cutting forces in a turning operation with flank wear (or clearance wear) to create a model of force-wear relationship. Similarly, Zhang et al. (2012) also compared and related cutting forces to tool wear to support that Minimum Quantity Cooling Lubrication (MQCL) was the optimal lubrication system for dry end-milling Inconel 718. On the other hand, other authors have used machine learning algorithms to create TCM systems with cutting forces. One example is the work of Wang et al. (2014), where they created a cutting forces-based TCM system using Relevance Vector Machine (RVM) and compared this to a Support Vector Machine (SVM). They developed a binary classifier RVM that delivered more accurate results than the SVM using several training samples. Furthermore, they also compared the classification processing time, finding that on average, the RVM was 35 faster than the SVM. Likewise, Azmi (2015) used cutting forces for an Adaptive Network-Based Fuzzy Interference Systems (ANFIS) to monitor tool wear in an end milling operation of glass fibre-reinforced polymer composites. This author compared two ANFIS models with different partitioning techniques (grid partitioning and subtractive clustering). The resulting models successfully matched the nonlinear relationship of tool wear and cutting forces, presenting low values of root mean square errors.

Also, several authors have taken different approaches to relate vibrations and tool wear, where some have also used machine learning for tool wear analysis and prediction. Orhan et al. (2007) used vibration as a general tool condition monitoring method during end milling, by finding threshold values that matched tool wear and vibration data. Furthermore, Möhring et al. (2016) also evaluated vibration signal in a milling operation to find its relationship with wear, proposing the integration of sensors inside the milling tool for future prediction work. There is also the work of Venkata Rao et al. (2013), who used a Taguchi design of experiments method to correlate tool wear with vibrations, implementing an ANOVA and regression analysis to evaluate tool life. On the other hand, inside the machine learning side of TCM, G. F. Wang et al. (2014) used a support vector machine (SVM) decision-making algorithm in order to compare C-SVM to v-SVM in a vibration-wear TCM system. Krishnakumar et al. (2015) also extracted vibration data from a high-speed machining operation of titanium alloy, to classify tool condition using a J48 Decision Tree and Artificial Neural Networks. As well, Sevilla-Camacho et al. (2015) successfully designed and implemented an online and real time TCM system, where digitised vibration signal was introduced into a single field-programmable gate array for training and monitoring of their prototype device.

On the side of acoustic emissions (AE), Kannatey-Asibu & Dornfeld (1982) showed an early use of AE for TCM, relating this signal to flank wear by finding that the kurtosis and skew of a β distribution for root mean square (RMS) acoustic emission signal is sensitive to chip contact to the tool's rake face and tool wear. Zhou et al. (2011) on the other hand, used AE to monitor tool life by implementing an auto-regressive moving average and compare results with cutting forces. Additionally Pawade & Joshi (2012) and Olufayo & Abou-El-Hossein (2015) used correlational methods to evaluate AE signal, where the first research included tool wear and workpiece surface roughness in the correlations, finding AE waveforms and amplitudes that correspond to different conditions and behaviours; the second work, however, only correlated AE data with tool wear, but set the basis for a proposed Neural Network for tool life prediction. Lastly, Ren et al. (2014) extracted AE signal data from micro milling operation to implement machine learning through a type-2 fuzzy logic system to monitor tool wear and estimate tool life.

While these articles showed a successful implementation of each monitoring method, some other authors found a similarly successful approach to TCM through a combination of methods. An example of this is the work carried out by Dimla & Lister (2000), where they analysed both cutting forces and vibrations to find their relationship with tool wear, proposing future work with neural networks. There is also the work of Haber et al. (2004) that included the use of cutting forces, vibrations and AE signal to monitor tool life. Ghosh et al. (2007) also combined these three signals to estimate average flank wear by implementing a neural network for tool wear prediction. And finally, Marinescu & Axinte (2008 & 2009) present two articles where AE are backed up by cutting forces signal to monitor tool wear and workpiece surface integrity in milling operations.

While all these approaches and methods are successful in evaluating, analysing and/or predicting tool wear, they are all highly dependent on the actual machining operation parameters and variables, namely: machining method, machining parameters, tool material and geometry and workpiece material. This is, therefore, an important disadvantage of indirect methods of TCM, however, when correctly implemented, they do perform a very accurate TCM in either a real time or an online manner.

The TCM system proposed in this research, being an indirect vision method, share this dependency on machining operation parameters and variables. However, the flexibility and effectiveness that vision could provide to a monitoring system are worth exploring. The wide availability of cameras and vision systems, along with their lowering costs, make these sensing devices more accessible. Furthermore, some of these methods require careful and accurate installation of devices inside the machining system. Cutting forces and vibrations sensors are commonly installed inside machine-workpiece fixtures, or even directly on the workpiece. Vision systems, on the other hand, can generally be installed almost anywhere inside the machining area, depending on their objective. Therefore, these systems are worth exploring, and different examples can be found in the next section.

2.2.1 Vision TCM Systems

Vision is another widely used method for both Direct and Indirect TCM. The access and costs of image acquisition systems have become much more accessible, and the computational capabilities of vision-based systems have improved significantly in the past decade. Furthermore, there are various imaging techniques that can be implemented for monitoring. The most widely used include cameras with CCD and CMOS sensors. These two types of sensors can efficiently capture and process a scene in the visible spectrum, either in intensity levels (grey-scale) or in full colour (RGB). The main difference between these is the processing speed, power consumption, light sensitivity, resolution and cost. CCD sensors generally have higher resolution and light sensitivity, with a reduced presence of noise, however, CMOS sensors are much less expensive and tend to have faster processing speeds and lower power consumption.

On the other side, there are other imaging techniques that deal with other multispectral spectral bands of light or hyperspectral bands of the electromagnetic spectrum, are mostly to accentuate or extract other non-visible features of a physical scene. This is the case of infrared cameras, which are widely used to capture temperature values. These devices are also broadly implemented in TCM, however, these are extremely expensive and slow processing cameras, and they deal with an entirely different approach than the one proposed in this research, as the analysis of the process temperature is out of the scope of this research. Similarly, imaging of other spectral bands could aid in accentuating certain features of the machining scene.

Another category of imaging techniques include modalities that are mostly used in medical imaging. These include X-Rays and Magnetic Resonance Imaging (MRI), where both can be captured in 2D or as 3D tomography. However, these techniques are very expensive and require configurations and resources that would be very difficult to implement in a machining environment.

These alternative techniques of multispectral and medical imaging methods would require an entirely different analysis and approach to the one proposed in this research. Therefore, the traditional visible spectrum imaging was selected for this research, but the use of other techniques could be part of this research's future work.

Many researchers have also worked on different approaches to monitor tool condition using traditional imaging, and in some cases, predict tool life. In Table 2.1 there is a selection of different authors that have worked with vision as a direct method of TCM.

Limiting to the literature regarding milling operations deemed relevant for the present research, it was found that most authors carried out direct imaging or measuring of cutting tool wear by either dismounting the tool or insert from the machine, or else doing it in-process. Xiong et al. (2011) presented a well-round and mathematically supported approach to tool wear measurement through an active contour detection methodology. They managed to accurately and repeatedly isolate tool wear by producing a binary image that includes only the worn region and where background noise was successfully and completely removed. However, their approach was based on complete interruption of the machining operation, removal of the cutting insert and use of a separate image capturing apparatus. This method, whilst allowing the capture of very clean and clear images, appeared to be highly dependent on the carefully calibrated illumination system; making this approach in-situ rather than in-process.

Wang et al. (2005), on the other hand, managed to develop an in-process system that managed to capture tool flank wear data inside the machine and while the spindle is rotating. They applied relatively more complex image processing techniques that included thresholding, segmentation and morphology operations, and used a fairly simple illumination system. The ability of doing the wear measurement on-the-fly is a very interesting and highly desirable capability, as avoiding a constant stopping of the machining can help preserve more continuous machine dynamics and improve machine health. As described in their article, this gives a high potential for industrial application.

Li et al. (2013) present a closer study to the one presented in the current research, exploring the measurement of tool wear for a milling process on Inconel 718, but using PVD coated carbides. They present an in-process image acquisition system to capture tool wear data with a simple illumination system. However, the measurement of tool wear was completely manual and without any complex manipulation of images through image processing. It could be argued that a manual assessment of tool wear could improve measurement accuracy, but this article presented only one test sample per material and no baseline with any other common method of wear measurement.

Finally, Wang et al. (2006) present in their article a very relevant approach to 3D mapping of crater wear through the use of a phase shifting method using fringe patterns. The article has a well-supported methodology, successfully capturing crater wear depth, with, centre distance and front distance with high accuracy; with high robustness to illumination variance and background layout. Nevertheless, a limitation identified for this technology is the selection of the fringe pattern, as its width impacts directly on the system's accuracy.

Table 2.1: Articles using vision as a direct method of TCM.

AUTHOR(S)	MACHINING	RESEARCH SUMMARY
Teshima et al. (1993)	Turning	Imaging of cutting tools and processing to extract flank and crater wear. Wear information and cutting conditions introduced as inputs in a Neural Network to estimate tool life.

Weis (1993)	Milling	Imaging of cutting tools and processing to extract flank and crater wear information.
Kurada & Bradley (1997a)	Turning	Processing of cutting tool images of flank wear, using texture-based segmentation. Proposes a simple, non-contact method of tool wear assessment.
Kurada & Bradley (1997b)	CNC machines	Review of the basic principles, instrumentation and processing techniques for vision-based TCM.
Pfeifer & Wiegers (2000)	Inserts	Discussion of advantages of machine vision as direct measurement technique. Gives general image processing techniques appropriate for tool wear measurement.
Lanzetta (2001)	Milling and Turning	Classification of tool morphologies and tool wear assessment, proposing a flow chart for defect recognition and quantitative assessment. Uses algorithms to recognise all the assessed defects implementing texture segmentation.
Devillez et al. (2004)	Turning	Measurement of tool crater wear using white light interferometry, in conjunction with cutting tool forces to catalogue tool wear and set the foundations of a monitoring system.
Wang et al. (2005) & Wang, Hong, et al. (2006)	Milling	In process system for flank wear measurement implementing enhancement, thresholding and segmentation in high-speed images, as the cutting tool is captured as it rotates.
Dawson & Kurfess (2005)	Turning	Measurement of volumetric loss in tool crater wear and flank wear, comparing results with cutting forces and machined surface topography for too life assessment.
Otieno et al. (2006)	Micro-milling	Measurement and classification of tool wear through Gaussian filters and histogram equalization, comparing used and unused tool profiles.
Wang, Wong, et al. (2006)	Milling	Measurement of crater wear by 3D image construction of cutting tool inserts using phase shifting fringe patterns.
Castejón et al. (2007) & Barreiro et al. (2008)	Turning	Imaging of flank wear and use of segmentation to find and classify geometrical descriptors. Use of Fowlkes-Mallows index along with other statistical analysis to evaluate wear evolution.
Alegre et al. (2009)	Turning	Digital imaging of tools' cutting edges, using contour signatures of wear region as input for classification and implementation of k-nearest neighbour and a neural network.
Xiong et al. (2011)	Milling	Images of cutting tools are processed to locate the wear area contour for tool wear measurement.
Zhang & Zhang (2013)	Milling	On-line measurement of flank wear in ball-end milling cutters, using subpixel edge detection to find tool wear area.
D'Addona & Teti (2013)	Turning	Images of cutting tools are processed to extract crater wear values for a neural network to estimate tool wear.

Li et al. (2013)	Milling	Integrate an optical system with a CNC machine to inspect and measure flank, nose and crater wear; and the results are compared to SEM-based measurements.
Zhu & Yu (2017)	Milling	Imaging of micro-milling tools and use of a region growing algorithm based on morphological component analysis to extract wear regions.

There is as well work regarding indirect methods of TCM, and a selection of these researches can be found in Table 2.2.

Table 2.2: Articles using vision as an indirect method of TCM.

AUTHOR(S)	MACHINING	RESEARCH SUMMARY
Bradley & Wong (2001)	Milling	Three techniques/parameters in the surface roughness assessment are taken for tool wear monitoring, using machine vision; these are histogram analysis, image frequency domain content and spatial domain surface texture.
Kassim et al. (2002) & (2006)	Turning & Milling	Fractal analysis of surface texture and implementation of Hidden Markov Model to classify various states of wear for TCM.
Kassim et al. (2004) & (2007)	Turning & Milling	Surface texture images are analysed for TCM using Hough transform, where in the 2004 publication, a multilayer perceptron neural network is applied to estimate flank wear.
Kang et al. (2005)	Milling	Depending on tool wear, a relation is found between surface roughness and fractal dimension, and it is used as an in-situ TCM system.
Bhat et al. (2016)	Turning	Images of the machined surfaces are processed using hidden Markov model, on features extracted from a grey level co-occurrence matrix, to classify and monitor tool wear.
Lee et al. (2016)	Turning	Through in-process imaging of workpiece’s surface roughness, cutting tool failure by chipping is monitored using autocorrelation analysis with subpixel accuracy.

It can be appreciated that many of the present articles on indirect measurement of tool wear through vision rely on the processing and analysis of surface roughness or surface topography. Most of these methods use different pattern recognition and machine learning methods to relate surface roughness to tool wear or to classify different stages of wear. However, a large percentage of these articles include a post-processing approach. Between these, most of their implementation was mainly to turning operations. Kassim et al. (2006), however, managed to implement fractal analysis of the surface texture to overcome the more variable surface roughness in milling operations. Kang et al. (2005) also used fractal analysis, however, they complemented the image data collected with a stylus type surface profiler. This methodology enabled them to capture a full surface topography that was later used to correlate this surface roughness to tool wear.

On the other hand, Bhat et al. (2016) managed to carry out the assessment of surface roughness in-process by locating a camera close to the machined workpiece inside the machine, acquiring images of the machined surface. They implemented a grey level co-occurrence matrix analysis, which can calculate spatial correlation properties of the surface texture, and later used a hidden Markov model for tool condition classification. Their approach appears to be an accurate on-machine tool wear monitoring system, however, the robustness of the apparatus presented in their article may be affected by variable illumination and cleanliness of the workspace area.

Lee, et al. (2016)also developed an in-process and potentially live tool chipping monitoring system through continuous assessment of workpiece surface roughness. This article includes a very well-round methodology and a very complete assessment of tool wear to evaluate tool chipping conditions and consequences. Using an autocorrelation sub-pixel analysis, the level accuracy of their technology appears high and their method quite robust. Their testing was close related to the present research, as their machining process used an aluminium oxide-based ceramic cutting tool on a hard to machine workpiece; and the apparatus was based in a common SLR camera.

Whilst these two tables and articles are just a sample of the work being carried out, it can be appreciated how the quantity of research in indirect vision methods of TCM is less when compared with direct vision methods, leaving an important gap in this area where further methods of indirect TCM can be explored.

2.3 Digital Image Processing

Digital image processing (DIP) refers to the acquisition of digital images which are then processed by a computer. DIP has grown considerably in the last decade due to the important advances made in the development of different algorithms (Cuevas Jimenez et al. 2010). Also, the constant reduction in costs of different image acquisition devices has made them very affordable for different tasks, such as the monitoring of a machining process. Whilst this last application will be further explored in the next section, the different principles of image processing that will be used for the research will now be explored and described.

A digital image can be defined as a two-dimensional function that quantifies light intensity (with the visible spectrum being the most commonly used). An image is commonly represented as *I* (*x* , *y*), where the intensity value is taken by the indexed coordinates *x* and *y*; and these intensity values are known in a digital image as *pixels*. The most common representation model of an image is given through an M X N numerical array or matrix, as shown in Equation 2.2 (Gonzalez & Woods 2010).

$$I(x,y) = \begin{pmatrix} I(0,0) & I(0,1) & \cdots & I(0,N-1) \\ \vdots & & \ddots & \vdots \\ I(M-1,0) & I(M-1,1) & \cdots & I(M-1,N-1) \end{pmatrix}$$

Equation 2.2

This digital image matrix can be then manipulated by different kinds of software for different purposes; MATLAB[1] is one of these, as it can work with these images' matrices and perform mathematical operations with them. This software was chosen for all the image processing carried out throughout the research, and for this reason, all the principles of DIP in the next sections will be oriented to their usage and representation in MATLAB, specifically to its Image Processing and Computer Vision toolboxes.

2.3.1 Grey Scale, Binary and Colour Images

There are three main types of image representation commonly used in image processing, these are:

1. Grey-scale or intensity images.
2. Binary or logical images.
3. Colour or RGB images.

Grey-scale or Intensity Images (see Figure 2.6) have 2D arrays of size M X N matrices, as mentioned previously, where each number of the matrix represents different values of intensity for a specific pixel. In MATLAB, each pixel has either an integer value in the range of [0,255] in the case of *unit 8* images, or else floating-point values scaled in a range of [0,1] for *double* images.

[1] MATLAB R2017a, Image Processing Toolbox and Computer Vision Toolbox, The MathWorks, Inc., Natick, Massachusetts, United States. URL: https://uk.mathworks.com/products/matlab.html

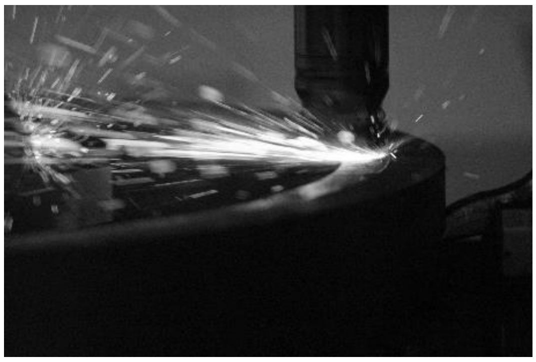


Figure 2.6: Grey-scale image of a machining process.

Binary or logical images (see Figure 2.7) on the other hand, have a logical array of 0s and 1s in their matrices. These types of images are built through a threshold value of intensity that separates pixel values into these two logical possibilities, displaying images in two colours, usually black and white.

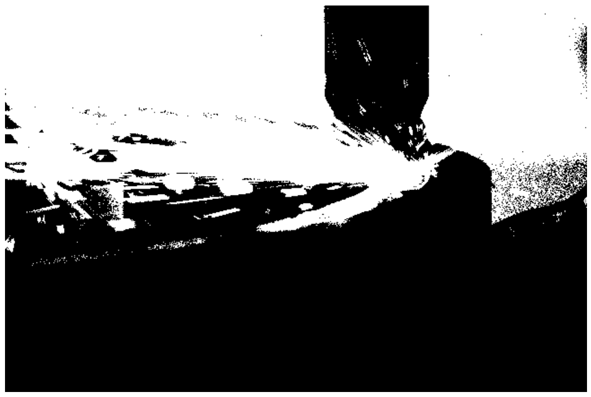


Figure 2.7: Binary image of a machining process.

Colour or RGB images are 3D arrays of M X N pixels or a stack of three grey-scale images, each with different intensity values. Each one of these matrices represents a colour channel in the resulting image: Red, Green and Blue. Colour images can also be represented with integer (unit8) and floating-point (double) values with ranges of [0,255] and [0,1] respectively (Cuevas Jimenez et al. 2010).

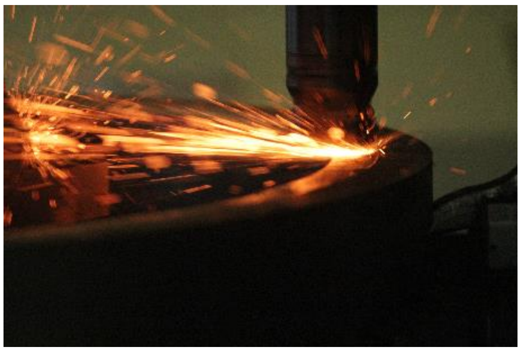


Figure 2.8: RGB image of the machining process.

These three formats are widely used for different purposes of computer vision and image processing, and MATLAB has different algorithms for image conversion between these formats. The algorithms of image conversion from colour images to grey-scale images (RGB to grey), and from grey-scale to binary images (grey to binary) are represented by Equation 2.3 and Equation 2.4 respectively.

$$I_g(x,y) = C_1 I_{RGB}(x,y,1) + C_2 I_{RGB}(x,y,2) + C_3 I_{RGB}(x,y,3)$$

Equation 2.3

$$bw(x,y) = \begin{cases} 1 \; if \; I_g(x,y) \le threshold \\ \quad 0 \; otherwise \end{cases}$$

Equation 2.4

In the RGB to grey conversion, $I_g(x,y)$ is the created grey-scale image and I_{RGB} is the original RGB image. The constants C_1, C_2 and C_3 give a weight to each colour component (Red, Green and Blue respectively) of the colour image, and in MATLAB their default values are C_1 = 0.2989, C_2 = 0.5870 and C_3 = 0.1140. In a basic grey to binary conversion, $bw(x,y)$ represents the binary image and the *threshold* discriminates intensity values to assign either 1 or 0 values.

2.3.2 Image Filtering

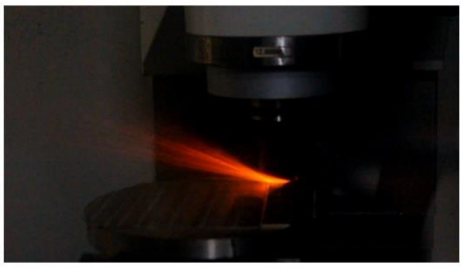
There are three basic filters that would be applicable for the isolation of the cutting sparks, these are low-pass, high-pass and band-pass filters. These filters are commonly used to either increase or reduce an image sharpness, nevertheless, they can also be used to reduce noise. They are normally applied to images in the *frequency domain*, using Fourier transformations (Cuevas Jimenez et al. 2010).

A *low-pass filter* H_{lp} has an *ideal* transfer function given by Equation 2.5. The term ideal indicates that all frequencies inside a defined circle radius are passed without attenuation, whereas all other frequencies outside are completely filtered out.

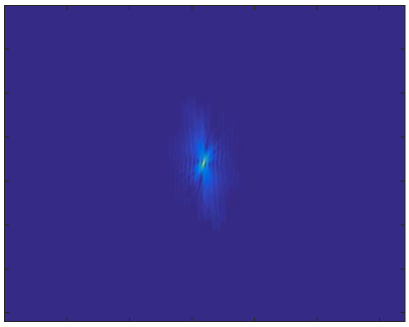
$$H_{lp}(u,v) = \begin{cases} 1 \; if \; D(u,v) \leq D_0 \\ 0 \; if \; D(u,v) > D_0 \end{cases}$$

Equation 2.5

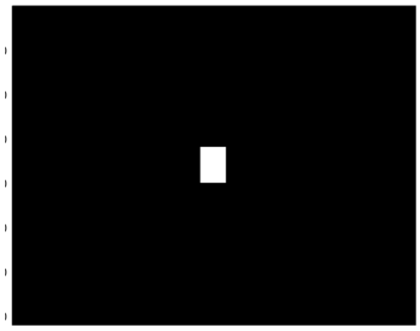
Where D_0 is a non-negative number and $D(u,v)$ is the distance from the point (u,v) to the centre of the filter, therefore only allowing frequencies inside the radius D_0 get through. In the present research, a rectangular shape of the low-pas filter was used and its visual representation is shown in Figure 2.9 (c). In Figure 2.9 (b), a shifted-absolute visual representation in the frequency domain of the cutting spark in Figure 2.9 (a) can be found. This visual representation shows the lowest frequency values in the centre, and increasing radially. Therefore, the low-pass multiplies by 1 the low frequency values inside the rectangular filter, and by zero all the rest, as shown in Figure 2.9 (d). The resulting image is shown in Figure 2.9 (e), where the image colour range was remapped to visualize the filtered image.



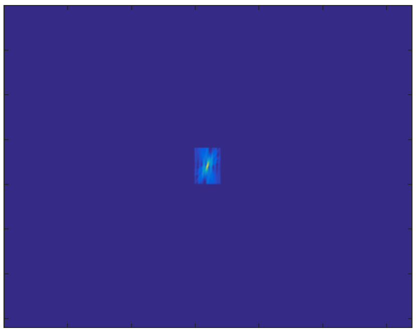
(a)



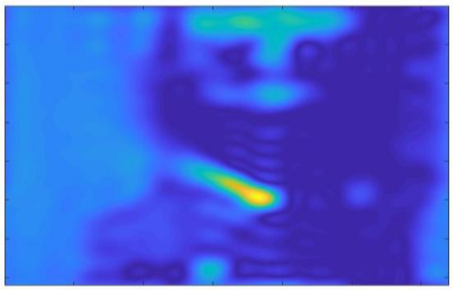
(b)



(c)



(d)



(e)

Figure 2.9: Original image (a), image in the frequency domain (b), ideal low-pass filter (c), result in frequency domain (d) and resulting image (e).

A *High-pass filter*, on the other hand, has the exact opposite structure to the low-pass filter, as shown in Figure 2.10 (a). For this reason, it can be expressed using Equation 2.6.

$$H_{hp}(u,v) = 1 - H_{lp}(u,v)$$

Equation 2.6

Where H_{hp} and H_{lp} are the high-pass and low-pass filters, respectively. This filter is also applied to the Fourier transformation of an image, as shown in Figure 2.10 (b).

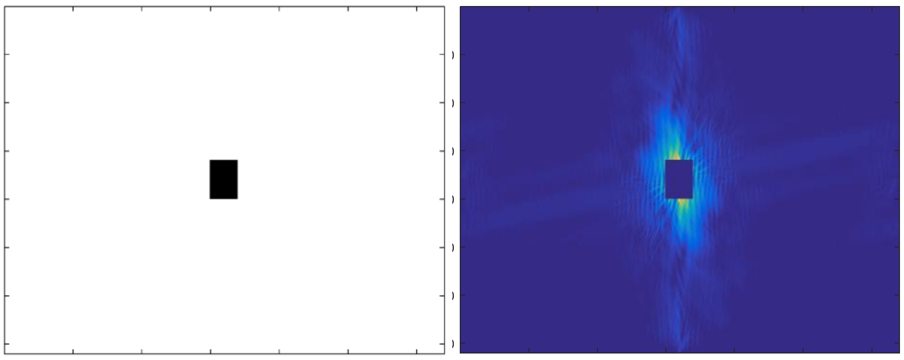


Figure 2.10: Ideal high-pass filter (a) and the result (b).

Finally, the *band-pass filter* is used to extract information in a certain direction (Figure 2.11 (a)), allowing either high or low values of frequency, depending on the application. Figure 2.11 (b) shows a horizontal band-pass filter, allowing through high values of frequency.

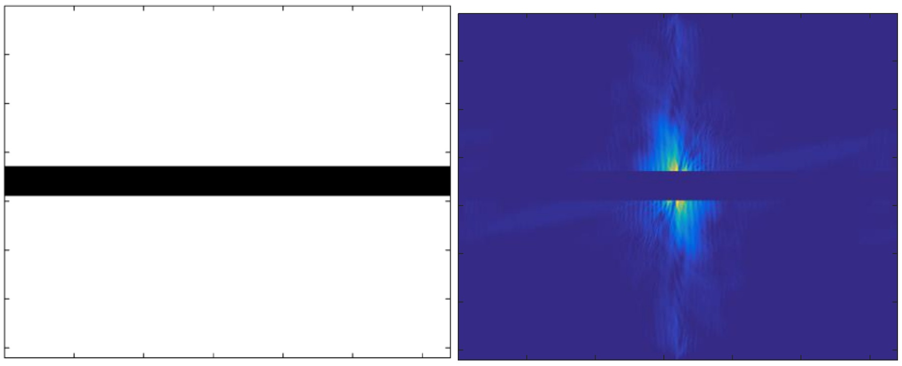


Figure 2.11: Ideal band-pass filter (a) and the result (b).

These filters can have different structures: they can have a circle or elliptical shape, instead of a rectangular one, or they can have a Gaussian distribution for a smooth transition from 0 to 1. These structures depend mainly on the type of application for these filters.

2.4 Artificial Neural Networks

In machine learning and pattern recognition, there are many models available for regression and classification of data. One of these are the Artificial Neural Networks (ANN), which are models that can effectively find relationships between datasets, even when their underlying rules are partially or completely unknown (Livingstone 2009).

ANN have their origins from the early 1940's to 1970's as an attempt to mimic the biological mechanism of the brain. During these initial stages of rapid development, neural networks demonstrated good results in pattern recognition as the first neuro-computing systems were built. However, exaggerated claims of the capabilities of ANN led to rejection from researchers at the end of this period. It was until the 1980's that the field of ANN experimented a resurgence as the model was revisited by many scientists and mathematicians, experimenting successful results as the capabilities of computer systems also improved, and resulting in a generalised renewed interest (Yadav et al. 2015).

The most popular model of ANN is the Multi-Layer Perceptron (MLP), which consists of a series of logistic regression models in the form of layers. This MLP generally include two layers of adaptive weights that are interconnected between nodes. Other models of machine learning, such as Support Vector Machine (SVM) and Relevance Vector Machine (RVM), present different approaches to the MLP. In the case of SVM, the objective function is convex-shaped, and it is based in the definition of basis functions. However, the number of basis function can become large, depending on the number of training points, which can become costly in processing time. RVM on the other hand, include a nonconvex optimization during training, and unlike SVM, it produces probabilistic outputs. Compared to these two, the MLP present an alternative approach, as it uses an adaptive parametric approach during training, resulting in a more compact and faster model (Bishop 2006). Consequently, the MLP was selected for this research and more information on MLP will be discussed in chapter 7.

2.5 Spark Formation

A wide search for literature on spark formation in machining operations was carried out, however no articles were found on this subject. Most articles found concentrate on different aspects of cutting chip formation and cutting mechanics, but none of them include an analysis or characterisation of spark formation. This absence of literature suggests a gap in this research area that could be quite relevant for the present research. Nevertheless, this will not be addressed in the present work, as it was believed it this could be an entire research project of its own.

2.6 Summary

This chapter attempted to give a broad idea of the literature that encompass and support the different approaches and decisions taken during the present research. Therefore, the chapter commenced with relevant information on ceramic cutting tools, regarding their composition and applications, leading to the material used in this research: SiAlONs. As described then, this material presents excellent properties for high-speed rough machining of nickel-based super alloys, giving an advanced effectiveness when compared to traditional carbide cutting tools. However, its effectiveness is somewhat limited by the quick way these materials get worn. Consequently, this was followed by literature that describes the different primary and secondary wear mechanisms that cutting tools experiment, followed by a summary of relevant articles that directly assessed SiAlON materials.

This led to the introduction of Tool Condition Monitoring (TCM), as literature describing the importance in understanding and managing tool wear and tool life was included. The different types of TCM systems (direct and indirect) and their respective methods were described, and the most relevant method for this research was introduced: Vision TCM Systems. Therefore, a summarised collection of some of the relevant research on direct and indirect vision-based TCM was presented. This had the objective of locating the present study into its respective area of research.

After these sections, some of the relevant knowledgebase found in literature of Digital Image Processing (DIP) was included to set the bases of the data processing tasks of the research. And finally, a short description of the historical and technical bases of Artificial Neural Networks (ANN) was included.

The following chapters are based in the information and literature presented in this chapter, as the proposed system of TCM is a novel approach, and therefore, no other literature that attempts to use cutting spark information to evaluate tool wear was found.

3 COMMON METHODOLOGY

As described briefly in the introductory chapter, the main purpose of the research concerns the implementation of a visual tool condition monitoring system. Such a system would, therefore, require the use of an image acquisition device, recording still pictures or video feed of the evolution of the cutting process's sparks. To analyse and select different aspects of the "optimal" method of image acquisition, and to explore other relevant aspects of this research, three main sets of experiments were carried out. In this chapter, these experiments will be described to lay out the foundations of the different studies and work included in the subsequent chapters.

3.1 General System Configuration

Even though different image acquisition devices, CNC machines and materials were used during the present research, they all followed the general system configuration shown in Figure 3.1.

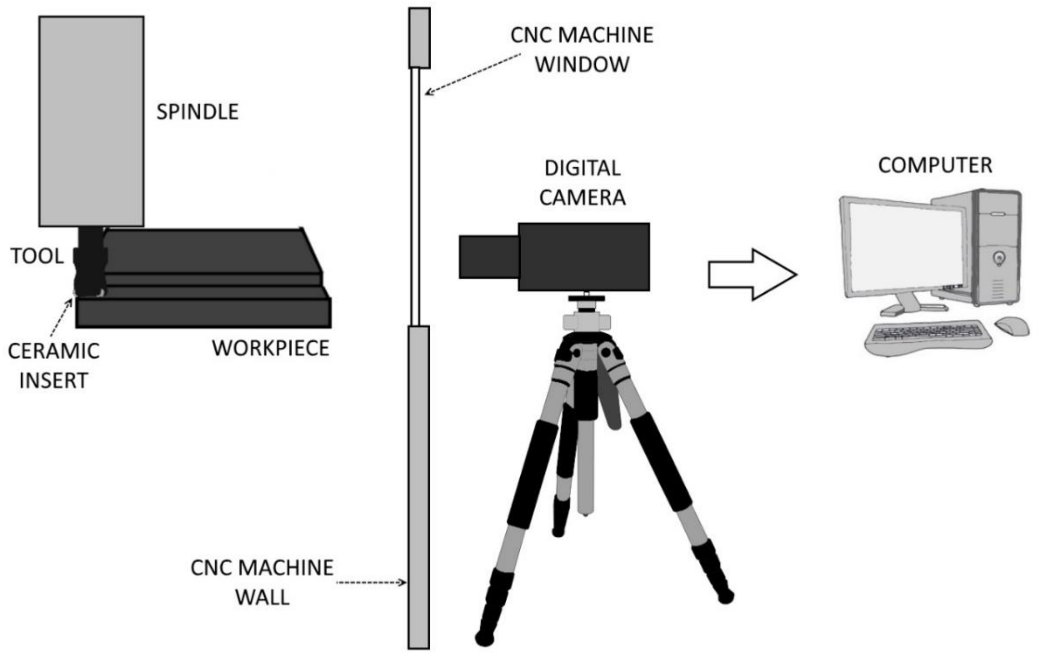


Figure 3.1: General system configuration.

It can be seen in the figure how this visual monitoring system was configured. It included the use of a digital camera or image acquisition device, recording images of the cutting area. The inserts tool holder used in the experiments can be seen in Figure 3.2.

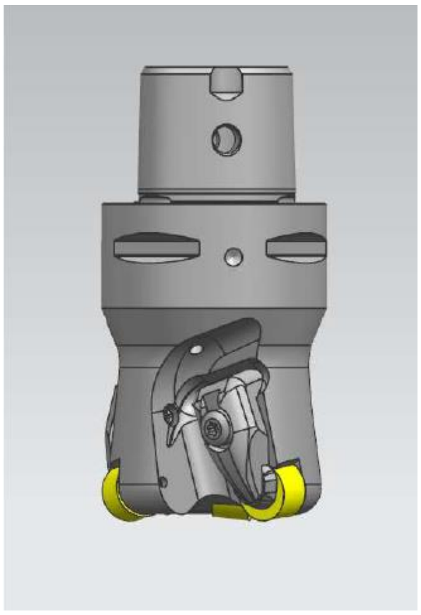
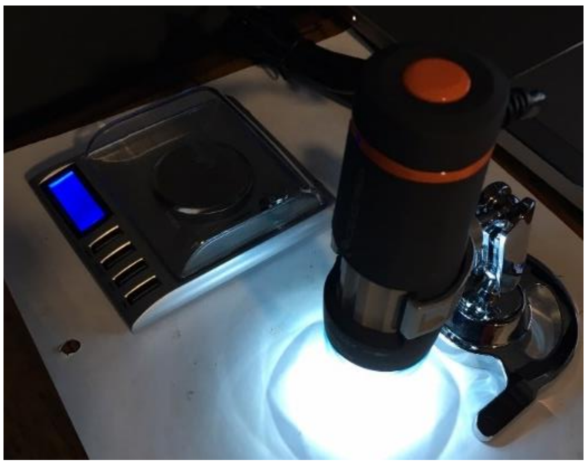


Figure 3.2: Round ceramic inserts tool holder by Sandvik.

The different parameters and configurations of the camera and general layout would assure the inclusion of as much visual contact with the entire cutting spark as possible. The digital camera had to be placed on the exterior of the machine due to its size and in order not to alter the general function of the CNC machine. In the author's opinion and experience with the different experiments, a recording device could also be placed inside the machine. This could improve the general stability and consistency of the image feed and would be ideal for a finalised system of visual tool condition monitoring (TCM). General lighting conditions were kept as constant as possible, however, due to the position of the machine in the workshop, there was still some variation of natural light. To mitigate the effect of this variable, the acquisition parameters selected in each test would attempt to obscure the image, so that the main source of light would come from the spark itself; this will be fully described in section 3.3.3.2.

After every session, the image data was sent to a computer, where image processing algorithms were used to extract relevant image features or spark descriptors. The software used for image processing was MATLAB (R2015b), which is a matrix-based platform used to manipulate computational mathematics in its own native programming language. It is also important to mention that the specific toolboxes used for this research are the "IMAGE PROCESSING TOOLBOX" and the "COMPUTER VISION SYSTEM TOOLBOX."

To obtain direct measurements of tool wear of the ceramic inserts, two additional tasks were performed in most of the experiments: insert weighing and insert digital microscope imaging, as shown in Figure 3.3 (a). By using high precision scales, it was possible to obtain mass loss data of the inserts after ever run (see section 4.2.2). This task was especially important to understand some of the tool wear mechanisms found in the cutting tools; this will be furtherly described in section 4.2.2. Also, the ceramic insert's crater and flank faces were photographed using a portable digital microscope, as shown in Figure 3.3 (b), after every run. Again, these images would be used to measure and assess different tool wear mechanisms.


(a)

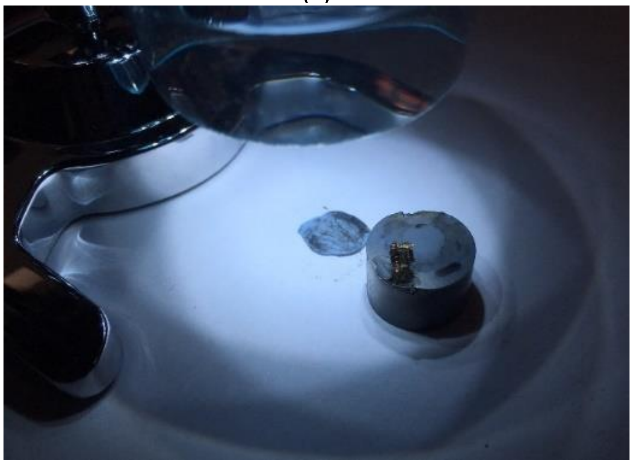

(b)

Figure 3.3: Additional tasks during experiments: (a) set-up, (b) digital microscope imaging.

3.2 Data Processing and Analysis

To extract the relevant data from the cutting process through the images acquired, spark descriptors had to be chosen to monitor and evaluate the change of spark evolution. Two main features were selected: spark intensity and spark area. These two were easy to identify, evaluate and analyse, being the most prominently changing features; while creating a smooth and very similar evolution to tool wear. Other descriptors such as spark angle, spark length and colourimetry were also analysed previously, however they did not remotely match the evolution of tool wear (Dominguez Caballero 2012).

Algorithms were written for the extraction of each spark descriptor, containing different operations relevant for each extracted feature (area and intensity). However, both these algorithms can be summarized by a general algorithm that was designed and segmented into the four stages shown in Figure 3.4.

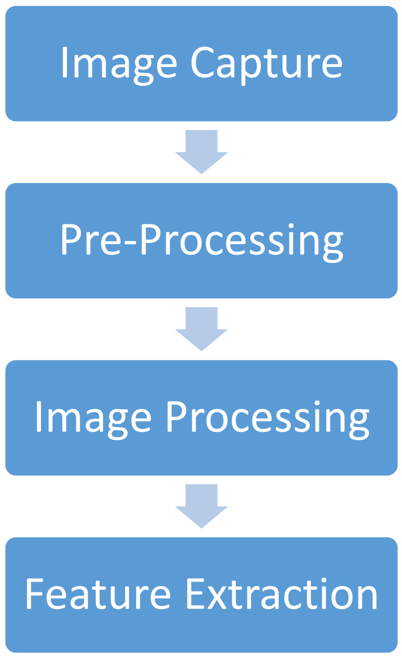


Figure 3.4: Feature extraction general algorithm segmentation.

The first stage of the algorithm included the *image capture* procedure, which varied between tests depending on the imaging device and type of feed recorded (still pictures or video). This stage was the same for both spark features.

3.2.1 Area Extraction

At the *pre-processing* stage, all images were downloaded into the algorithm in the form of three-dimensional matrices for RGB images, and one-dimensional matrices for the greyscale images. Still pictures were read directly by a sequential loop and into a multidimensional array. In the case of RGB images, they were stored in a MATLAB's structure array, and greyscale images were stored into a k-dimensional variable ($\mathrm{m} \times \mathrm{n} \times \mathrm{k}$); where m is the height of the pictures in pixels, n is the width in pixels and k is the total number of images.

In the case of the video feed, a section for frame extraction was included in the algorithm, before the mentioned image storage. This section would read the video file and use a loop to separate it into its respective frames. These frames could be then treated as still pictures and hence proceed to the previously mentioned image storage routine.

Even though it could have potentially simplified the image processing tasks, it was decided not to do a manual cropping of the images to isolate the area of interest. Instead, a more unsupervised system of spark detection and isolation of the full image was implemented. Having a full field image helps acquiring a wider view of the sparks and its variation. Also, this unsupervised approach is related to the future of the research into a potentially fully automatic and unsupervised system.

After these tasks, the first picture of each compound was stored as a *reference image*, where the idea was to select an image with all the background layout, but with almost no visible spark.

Then, the *image processing* stage started with the matrix subtraction of the mentioned reference image R from each one of the images I_k (Figure 3.5) and creating a resulting subtracted image I_{subt}. This task had the purpose of eliminating the background information using Equation 3.1, and as shown in Figure 3.5 (b), where the area of interest is encircled. As there were irrelevant bright sections in the image that could be wrongly isolated by the algorithm instead of the cutting spark, this step assured that high intensity areas would come from the spark itself; therefore, removing noise from illumination inconsistency.

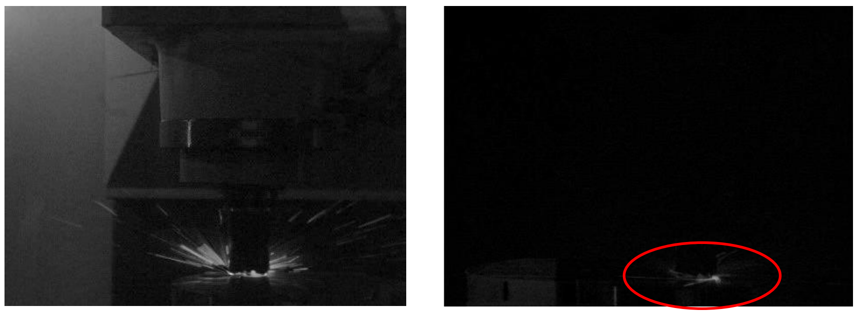

(a) (b)

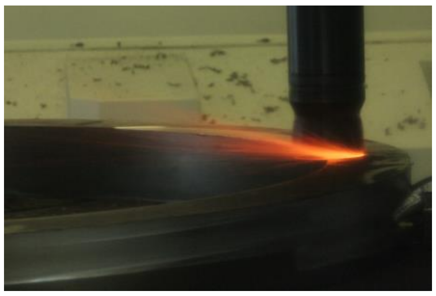
Figure 3.5: Result of image subtraction: original image (a) and resulting image (b).

$$I_{subt}(k) = I_k - R$$

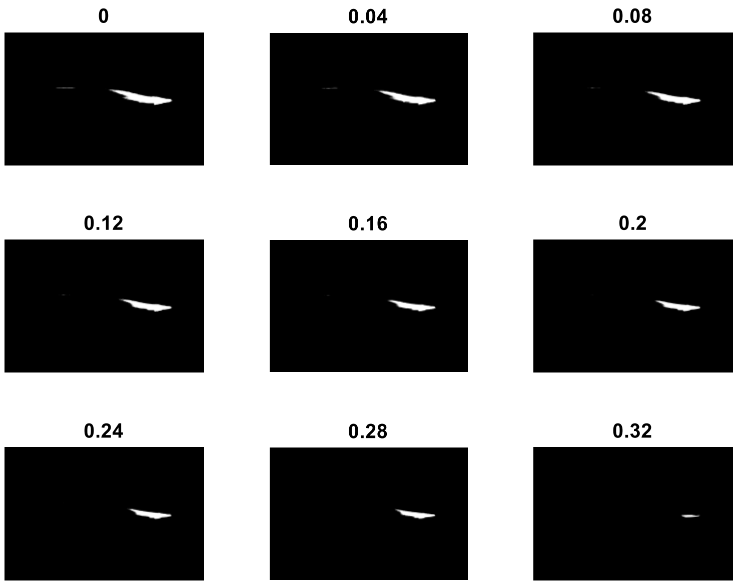
Equation 3.1

After this, the algorithm performed image enhancements and conversions. In the case of greyscale images, enhancements consisted in re-mapping intensity values of certain low and high values into a new image. In this way, only the intensity values belonging to the spark would be enhanced, making an appropriate spark isolation. In the case of RGB images, a re-mapping of intensity values was also performed, however, this was applied to each colour channel (red, green and blue). Re-mapping enhancement was found to be more effective in RGB images, rather than in greyscale images; given that the cutting spark contained very specific colours, improving the effectiveness of the spark isolation. Later, the mentioned image conversions consisted of converting images into greyscale, in the case of RGB, and then into binary images, in both cases. The greyscale conversion was performed using a standard MATLAB command, which uses Equation 2.3 in section 2.3.1 to weight each colour (R, G and B).

The binary conversion was performed through the luminance threshold *level* in the range [0,1] previously shown in Equation 2.4 in section 2.3.1. Figure 3.6 (b) shows an array of binarizations of the original image (a), displaying on the top of each image the threshold value used for the conversion. For each experiment and algorithm, the threshold value used was selected visually, choosing the image that included the largest portion of the spark whilst introducing the least noise.



(a)



(b)

Figure 3.6: Binary conversion (a) original image, and (b) binary conversion threshold selection.

Through this logical format of the image, it was easier to identify and isolate elements in the image, such as the spark. However, the presence of noise was an ever-present problem, as demonstrated by the encircled areas in Figure 3.7.

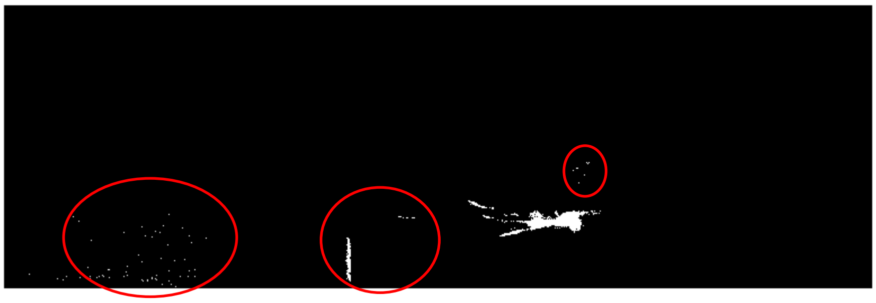
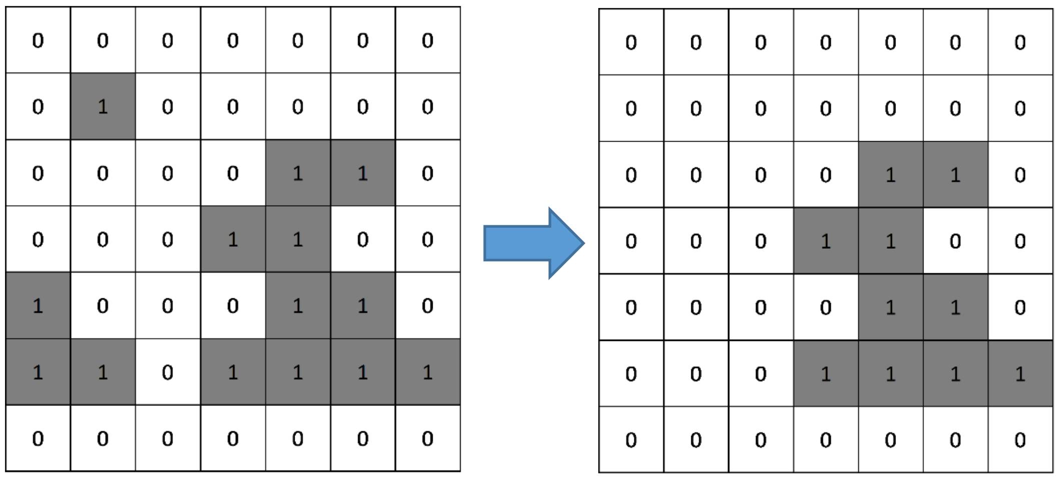


Figure 3.7: Binary image with examples of noise.

The next step was then to isolate the area of interest even further by image segmentation, implementing *connected components* filtering. Connected components are pixels that when centred in a 3 x 3 window, they have at least another non-zero component around. Connected components filtering consists in using a threshold value to define the minimum quantity of pixels that will be considered as an *image element* (Gonzalez & Woods 2010). Figure 3.8 illustrates this method of filtering, where only image elements that have 4 (threshold) or more than connected components are allowed through. Therefore, the image on the right only includes the largest image element with 10 connected components, eliminating the other two. Inside the present area algorithm procedure, the threshold value was selected to define the minimum quantity of pixels regarded as a spark, assigning zero values to components below this number.



0	0	0	0	0	0	0
0	1	0	0	0	0	0
0	0	0	0	1	1	0
0	0	0	1	1	0	0
1	0	0	0	1	1	0
1	1	0	1	1	1	1
0	0	0	0	0	0	0

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	1	1	0
0	0	0	1	1	0	0
0	0	0	0	1	1	0
0	0	0	1	1	1	1
0	0	0	0	0	0	0

Figure 3.8: Connected components filtering.

The final stage of *feature extraction* consisted of computing the spark area in each image. Given that all previous steps were oriented to isolate the spark, it can be concluded that the sole component in each image was the spark. Therefore, Equation 3.2 shows how the spark $Area$ was computed, where $cc[i,j]$ is the resulting image after connected components filtering. Given that the image is binary, with logical values with dimension [0.1], the total value of this summation will be equivalent to the total number of pixels in the spark.

$$Area = \sum_{i=1}^{n} \sum_{j=1}^{m} cc[i,j]$$

Equation 3.2

3.2.2 Intensity Extraction

For the extraction of intensity, the general algorithm shown before in Figure 3.4 was also applicable. The initial stage of image production and pre-processing included the exact same structure as in the area extraction. It is, however, in the image processing and feature extraction stages that some different processing techniques were used.

In the image processing stage, the same image subtraction and image enhancement process as in the area extraction was used. However, to isolate the intensity information of the spark, a filtering technique was introduced and a low-pass filter was selected. A description of different filters can be found in the literature review in section 2.3.2. An analysis of these filters and parameters was conducted in previous work, and a low-pass filter was found to be an appropriate option for the present application. This was concluded given that the spark would generally be a solid and consistent element in the image, including low-frequency values. High-frequency elements on the other hand, generally comprise of repeated and drastic changes in intensity, which is the case of edges and noise. Therefore, while low pass filters tend to blur or even decompose an image, in the present case the high, and to a certain degree, constant intensity inside the spark is successfully isolated (Dominguez Caballero 2012).

To select the size of the low-pass filter window, a sub-task compared the lowest frequency value found in the centre of image in frequency domain with its surrounding values in a horizontal and vertical line. If a surrounding value exceeded a certain threshold, this value's position would describe an edge of the filter window. This threshold was derived from the same lowest frequency so it could be adapted to different images; and it was selected experimentally by trying different levels and assessing results.

Accordingly, the algorithm implemented a low-pass filter to each image in the frequency domain by a Fourier transformation $\mathcal{F}$, as shown in Equation 3.3.

$$B = \mathrm{I} \otimes \mathcal{F}^{-1}(H_{lp})$$

Equation 3.3

Where B is the resulting image, I is the image processed to this point in the algorithm and $H_{lp}$is the low-pass filter. In Figure 3.9 (a) - (d), the implementation of this equation can be appreciated.

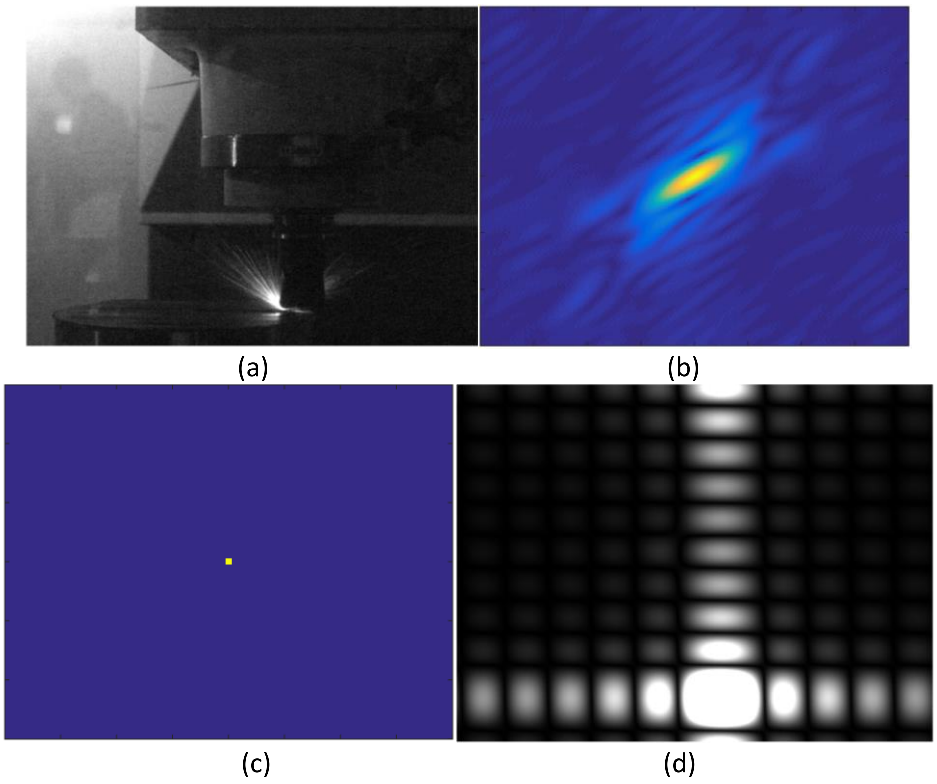


Figure 3.9: Original Image (a), image in the frequency domain (b), a low-pass filter (c) and resulting image (d).

Figure 3.9 (d) displays how the image intensity is now governed by the spark. Figure 3.10 shows a series of images taken during a milling operation and their respective resulting images when a low-pass filter was applied (Dominguez Caballero 2012). The resulting images show more detail of the original background than Figure 3.9 (d) because a wider low-pass filter window was used to illustrate the filter interaction with the original image. However, the intensity of the spark again governs the image intensity.

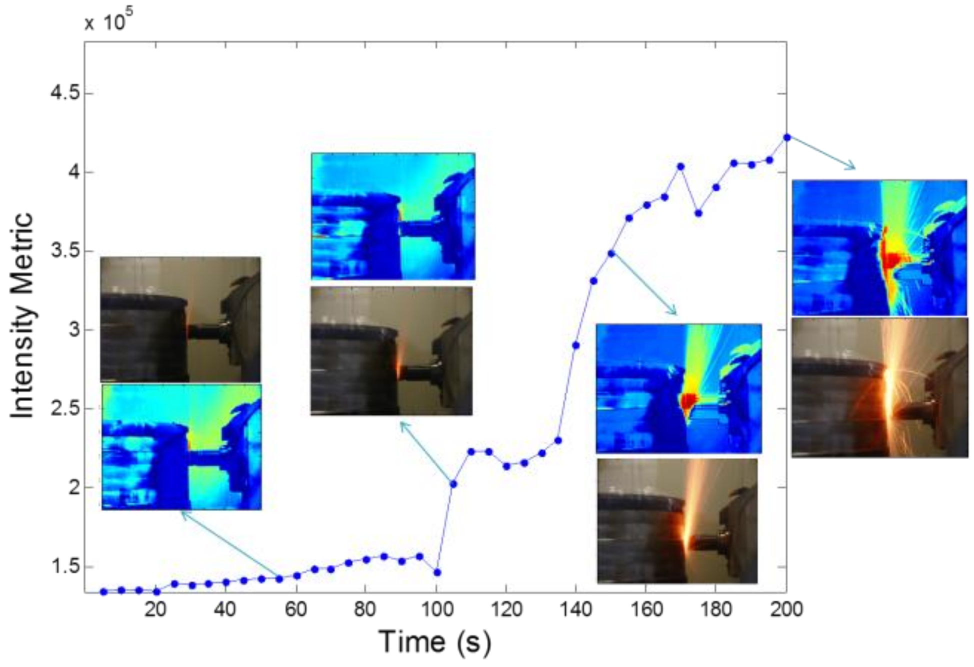


Figure 3.10: Images from a milling operation including their respective low-pass filtered resulting image, and intensity level (background graph) (Dominguez Caballero 2012)**.**

Therefore, at the feature extraction stage of the algorithm, a summation of all the elements in the image matrix B was performed to calculate total $Intensity$, as shown in Equation 3.4.

$$Intensity = \sum_{i=1}^{n} \sum_{j=1}^{m} B[i,j]$$

Equation 3.4

3.3 Experiments

As mentioned at the beginning of this chapter, three main experiments sessions were carried out during this research. This was the maximum number possible, given the availability of materials and resources, as well as the accessibility to machine time greatly reduced the possibility of performing more tests. However, the experiments carried out were successful, the specific session objectives were achieved, and the data obtained was adequate.

In this section, the three main experiment sessions will be described, including the materials, parameters and devices used. Nevertheless, all the experiments will be revisited in the upcoming chapters to describe in more detail their planning and the selection of their various resources, as well as to support the different analyses carried out in each chapter.

3.3.1 First Experiments – Previous Work Revisited

It was stated in the introductory chapter, in the background section, that this research follows on from previous work, initiated at the author's Master's degree. In that study, the main results were the successful extraction of spark intensity, analysing different filtering approaches, and the first approach to comparing spark evolution with actual tool wear.

In the present research, on the other hand, the initial objectives were to create a spark area extraction approach and to improve the algorithms already created in the previous work. For this reason, these first experiments describe the testing parameters used in that previous work, as they would be revisited and reused to lay some of the foundations of this research in the chapter 6. It is important to mention that these tests were carried out at the University of Sheffield's Advanced Manufacturing Research Centre (AMRC) as part of a different project, therefore, the machining parameters were preselected for that project.

This initial monitoring system was built using a regular SLR Canon EOS 60D digital camera, set to record images of the cutting area of the machining process, as shown in Figure 3.11. The milling machine used was a Starrag ZT 1000 5 axis CNC with a Siemens Sinumerik 840D controller. Four tests were carried out and during the machining time, the camera captured images at a constant rate, with a special focus on the sparks that this process produced. These images were later transferred from the camera to a computer to implement the different image processing algorithms.

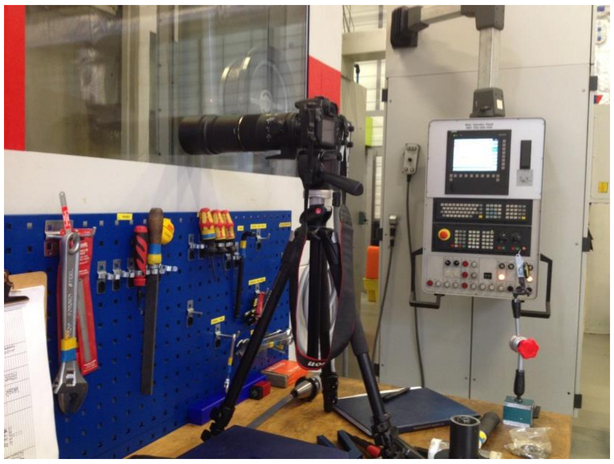


Figure 3.11: First set of experiments system set-up at the workshop.

3.3.1.1 Image Acquisition

In contrast to the human eye or a video feed, still pictures are a single sample of a short period of time. In an SLR camera, the shutter speed parameter is the one that controls the sample length in a single image. Therefore, the selection of speed parameters can be a common dilemma that is mainly dependent on the application. In Table 3.1 the parameters of the four tests can be appreciated, where these parameters were categorised as fast, medium and slow. Parameters in the third and fourth tests were the same given some conclusions that were drawn at that point of the research. Additionally, the fourth test included machine stops between 8 runs for tool wear measurements. The first objective of these tests was to assess which of these parameters were the most appropriate for this application using test one, two and three, discussed in the chapter 5. The second objective, discussed in the chapter 6, was to support the research hypothesis of a spark-wear relationship using test four.

Table 3.1: First experiments tests using fast, medium and slow shutter speeds.

	TEST 1	TEST 2	TEST 3	TEST 4
PARAMETER	FAST	MEDIUM	SLOW	SLOW
Speed	1/640	1/125	1/5	1/5
Aperture	F3.2	F5.0	F22	F22

3.3.1.2 Machining and Parameters

The study was performed using a five-axis high-speed milling machine and all of the tests used the machining parameters shown in Table 3.2. The cutting tools used were SiAlON ceramic inserts RNGN120400E 6060 from Sandvik Coromant, in a 4 inserts and 63 mm tool holder. The workpiece was a Waspaloy ring (Figure 3.12) and the general standardized composition of this material is shown in Table 3.3.

Table 3.2: Milling machining cutting parameters.

PARAMETER	VALUE
Cutting Speed *Vc* (m/min)	875
Feed per minute *Vf* (mm/min)	1843
Spindle speed *n* (rpm)	4761
Cutting Depth *ap* (mm)	1.5
Tool Diameter *D* (mm)	63
Radial Immersion *ae* (mm)	29.25
Feed per tooth *fz* (mm)	0.097
Number of inserts *zc*	4

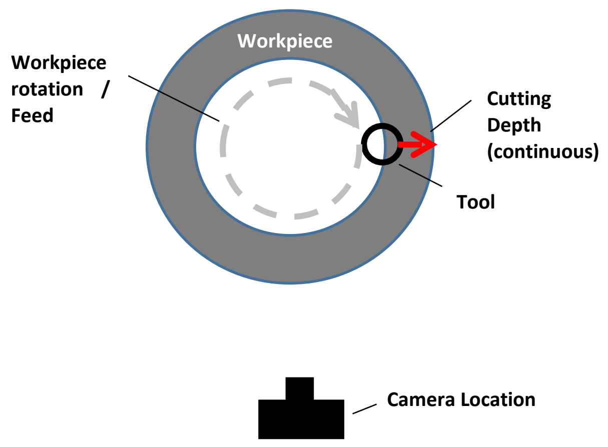


Figure 3.12: Waspalloy workpiece ring.

Table 3.3: Standardized waspalloy composition.

ELEMENT	MIN %	MAX %
Carbon	0.02	0.10
Manganese	--	0.50
Silicon	--	0.75
Chromium	18.0	21.0
Nickel	Balance	
Boron	0.003	0.008
Iron	--	2.00
Cobalt	12.0	15.0
Titanium	2.60	3.25
Aluminium	1.00	1.50
Molybdenum	3.50	5.00
Zirconium	0.02	0.12
Copper	--	0.10
Sulphur	--	0.02

3.3.2 Second Experiments – High-speed Feed

To fully assess the impact of image acquisition parameters in this application, this second set of experiments included the use of a high-speed feed. Inside these experiments, two tests were carried out using two different sets of machining parameters. Their results and analysis can be found in the chapter 5.

In this new configuration, a Phantom v210 high-speed camera was used to record grayscale video feed, using a Nikon AF 24-85mm f2.8-4 D IF lens. The camera manufacturer's software (PCC 2.5.744.0) was used to download the video feed and to extract images from each video frame. The milling process was carried out using a 3 axis XYZ 1060HS VMC milling machine with a Siemens Sinumerik 840D controller. Again, Sandvik ceramic SiAlON inserted tools RNGN120700E grade 6060 were used, with a 50mm S-R120R-038C5-12x03 inserts tools holder and a C5-390.55-40 030 attachment to fit into the CNC machine. The workpiece material in this new test was an Inconel 718 round block of 254 mm in diameter, where a 197 mm x 197 mm square was machined for tests consistency, as shown in Figure 3.13. The CNC programme used to machine this square can be found in Appendix I. Furthermore, this material was provided by the Mexican Forging Company FRISA and its general composition can be found in Table 3.4.

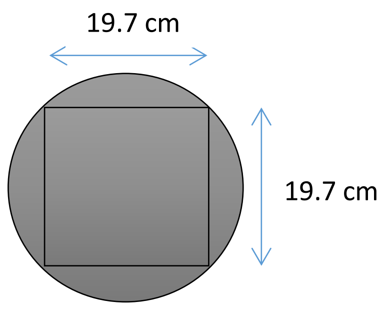


Figure 3.13: Inconel 718 workpiece.

Table 3.4: Inconel 718 general composition.

ELEMENT	MIN %	MAX %
Carbon	-	0.08
Manganese	-	0.35
Silicon	-	0.35
Phosphorus	-	0.015
Sulphur	-	0.015
Chromium	17.00	21.00
Nickel	50.00	55.00
Mo	2.80	3.30
Niobium	4.75	5.50
Titanium	0.65	1.15
Aluminium	0.20	0.80
Cobalt	-	1.00
Tantalum	-	0.05
Boron	-	0.006
Copper	-	0.30
Lead	-	0.0005
Bismuth	-	0.00003
Selenium	-	0.0003
Iron	balance	

3.3.2.1 Image Acquisition

The high-speed camera was used to obtain video feed from the cutting process, enabling later extraction of frames as individual images for future processing. The camera and video software settings used during the experimental sessions are shown in Table 3.5. These settings were primarily selected due to the functionality of the high-speed camera, as only specific configurations enabled the recording time required for each run. Inside these configurations, the resolution was selected after different trials, as it presented a good and clear image size that enabled a shorter processing time. The sample rate was selected as the fastest sample rate possible when selecting the image size and exposure time (this relation is constrained by the software). This last parameter and the aperture were selected after trials with different configurations. The combination selected of 3300 µs and aperture of 5.6 provided an adequate general illumination in the images for processing.

Table 3.5: Camera and video software settings.

SOFTWARE	
PARAMETER	VALUE
Resolution (Pixels)	800 x 600
Sample Rate (Frames per second)	300
Exposure time (µs)	3300
CAMERA	
PARAMETER	VALUE
Aperture	5.6

3.3.2.2 Machining and Parameters

For the two experimental sessions that were carried out, it was decided to use only one ceramic insert per session, Figure 3.14.

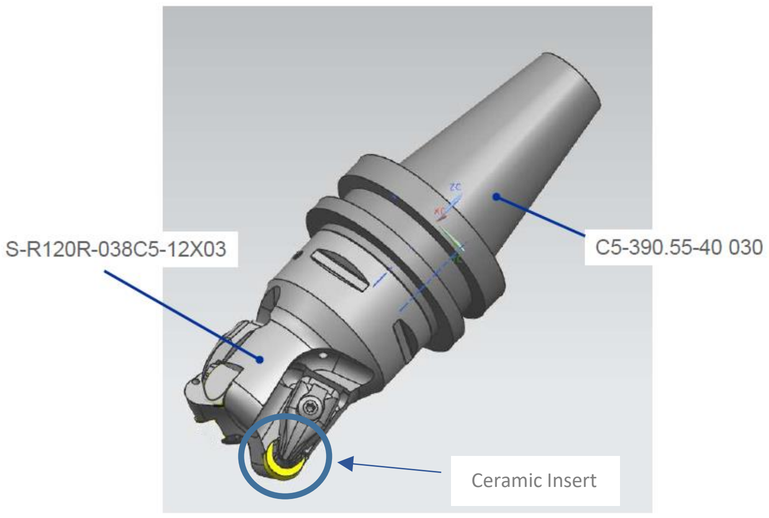


Figure 3.14: Single Insert tool configuration.

By using only one insert, the wear data can be isolated, facilitating data acquisition from spark evolution. The chosen cutting speeds (Vc), feeds per tooth (fz), cutting depth (ap) and radial immersion (ae) were recommended by the machining section of the Sandvik's

Ceramics Application Guide (Sandvik Coromant 2010). Afterwards, their corresponding feed speed (Vf) and spindle speed (n) were calculated. The parameters for both experimental sessions are shown in Table 3.6.

Table 3.6: Milling machining cutting parameters for Test 1 & 2.

PARAMETER	TEST 1	TEST 2
Feed per minute *Vf* (mm/min)	620	636.62
Spindle speed *n* (rev)	6200	6366.2
Cutting Speed *Vc* (m/min)	973.89	1000
Feed per tooth *fz* (mm)	0.1	0.1
Cutting Depth *ap* (mm)	1.5	1.5
Number of inserts *zc*	1	1
Radial DOC (%)	77.04	77.04
Tool Diameter *D* (mm)	50	50
Radial Immersion *ae* (mm)	38.52	38.52
Number of Runs	8	5

The Inconel workpiece was up-milled horizontally, performing 5 runs per level of material, as shown in Figure 3.15. The solid black circle represents the cutting tool, machining horizontally from left to right, through the dash-lined arrow, on each run. The other four dash-lined circles represent the next positions of the tool for the rest of the runs.

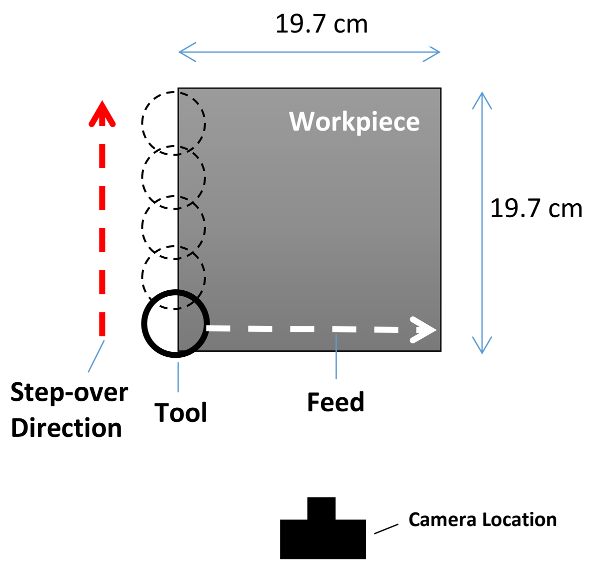


Figure 3.15: Runs sequence and orientation.

Furthermore, between runs in Test 1, the ceramic insert was extracted from the tool to be weighed and photographed. In Test 2 the insert was also photographed, but in-situ, whilst still mounted into the tool, and in this test, no mass measurements were extracted.

3.3.3 Third Experiments – Final Set

The analysis of the optimal image acquisition system and parameters can be found in the chapter 5, where still pictures from the first test and the high-speed video feed from the second test were compared. Conclusions drawn are the main justification for the selection of devices and parameters implemented in this final set of tests. In summary, the first experiments showed better results while using slow image acquisition settings. Later the second experiments proved that indeed, fast settings delivered noisy and unsteady results. This analysis also proved that the use of a common video feed device would deliver more data than a still picture device; and that in conjunction with an *image combination* algorithm, it would be possible to control and emulate slow imaging settings.

Therefore, this final set of experiments included again the use of a Canon EOS 1200D Digital SLR Camera with an EF-S 18-55 mm f/3.5-5.6 III Lens. The general test configuration was again the same as described at the beginning of this chapter, having the camera outside the CNC milling machine. Furthermore, and such as in the previous experiments, a video feed was acquired instead of still pictures. Similarly, the CNC milling machine, cutting inserts and tool holder were the same ones used in the second set of experiments, as well as the workpiece material and dimensions. However, in the final set of experiments, a *Design of Experiments (DOE)* was implemented meaning that the machining parameters were varied from test to test.

3.3.3.1 Design of Experiments for Machining Parameters

Designs of experiments (DOE), as its name dictates, is a method used to plan a set of experiments, selecting the appropriate data for further analysis, generally giving valid and objective conclusions (Mandal et al. 2011). DOE is mostly used to find relationships between input variables or parameters and output performance in an experiment (Antony 2003). Even though some correlations between tool wear and machining parameters will be discussed later in this thesis, the main objective of using a DOE was to plan these new tests in a way that different combinations could be tested.

Some widely-used methods of DOE are factorial designs and the Taguchi method. Full and fractional factorial designs are the most commonly used experimental designs in manufacturing. However full factorial designs tend to require a large quantity of tests, as all the possible combinations of input variables or factors, and their respective values or levels are tested. Fractional factorial and Taguchi methods, on the other hand, can greatly reduce the number of tests, whilst delivering enough data. However, in the present research, it was decided to use Taguchi methods of orthogonal arrays, given that fractional factorial designs can leave interactions between variables undetermined.

Taguchi methods have been widely used in machining research (Mandal et al. 2011; Ghani et al. 2004), as it allows researchers to accurately find correlations between cutting parameters and different outputs, such as surface finish, chip removal rate, wear, etc. These are especially important to optimise procedures and find the best combinations of parameters for specific tasks. Taguchi methods are based on orthogonal arrays that have a specific structure that provides justification for the selection of variables for each test. In these arrays, the possible pairs of factors are all explored the same number of times, as shown in Table 3.7. This table shows the orthogonal array denoted as L_9, with four factors (A, B, C and D) and three levels (1,2 and 3), organised into only nine tests, instead of the 81 tests required by a full factorial. Therefore, this method reduces the number of tests, but successfully explores every possible relation between these pairs of factors and levels (Kacker et al. 1991).

Table 3.7: Orthogonal array for L9 DOE using Taguchi.

TEST NO.	FACTOR A	FACTOR B	FACTOR C	FACTOR D
1	1	1	1	1
2	1	2	2	2
3	1	3	3	3
4	2	1	2	3
5	2	2	3	1
6	2	3	1	2
7	3	1	3	2
8	3	2	1	3
9	3	3	2	1

For the present research, it was deemed relevant to obtain a good coverage of data for the approach presented in chapter 7, which is an important advantage of using DOEs. Furthermore, given that the relation between machining parameters and tool wear may not be linear, three levels were chosen assuming a possible quadratic relationship. This would also impact in chapter 7, allowing a curved hyperplane of the design neural networks.

The four factors and three levels each are shown in Table 3.8. These factors and levels were in accordance to the window of parameters that Sandvik Coromant recommend in their manual of Ceramic tools, in the section concerned with ceramic milling (Sandvik Coromant 2010).

Table 3.8: Factors and levels selected for DOE.

FACTORS	LEVEL 1 (LOW)	LEVEL 2 (MEDIUM)	LEVEL 3 (HIGH)
Cutting Speed (Vc)	700 m/min	850 m/min	1000 m/min
Feed (fz)	0.07 mm	0.09 mm	0.11 mm
Cutting Depth (ap)	1 mm	1.5 mm	2 mm
Radial Immersion (DOC)	50%	70%	90%

Given that the selected parameters include the exact same number of factors and levels as in Table 3.7, the L_9 structure could have been used. However, given that the objective of the present experiments was to obtain training and verification data for a neural network (see chapter 7), it was decided to double the number of tests into an L_{18} array of 18 tests, generated from two L_9 arrays. Therefore, these parameters were introduced into the statistical software SPSS Statistics to generate the orthogonal array shown in Table 3.9.

Table 3.9: Orthogonal array L18, generated using SPSS Statistics.

TEST NUMBER	CUTTING SPEED Vc (m/min)	FEED fz (mm)	CUTTING DEPTH ap (mm)	RADIAL IMMERSION (%)
1	1000	0.09	1.5	70
2	700	0.09	1	70
3	700	0.09	2	90
4	850	0.11	2	70
5	1000	0.07	2	90
6	850	0.09	1	50
7	1000	0.11	1	90
8	700	0.07	1	50
9	700	0.07	1.5	90
10	700	0.11	1.5	50
11	1000	0.07	1	70
12	850	0.07	1.5	70
13	1000	0.09	2	50
14	1000	0.11	1.5	50
15	700	0.11	2	70
16	850	0.09	1.5	90
17	850	0.11	1	90
18	850	0.07	2	50

This array, for neural network implementation, could be divided into two L_9 arrays A and B, as shown Table 3.10 and Table 3.11; each table for training and verification respectively.

Table 3.10: L9 array A, extracted from the generated L18 array.

TEST NUMBER	CUTTING SPEED Vc (m/min)	FEED fz (mm)	CUTTING DEPTH ap (mm)	RADIAL IMMERSION (%)
8	700	0.07	1	50
2	700	0.09	1	70
10	700	0.11	1.5	50
12	850	0.07	1.5	70
16	850	0.09	1.5	90
4	850	0.11	2	70
5	1000	0.07	2	90
13	1000	0.09	2	50
7	1000	0.11	1	90

Table 3.11: L9 array B, extracted from the generated L18 array.

TEST NUMBER	CUTTING SPEED Vc (m/min)	FEED fz (mm)	CUTTING DEPTH ap (mm)	RADIAL IMMERSION (%)
9	700	0.07	1.5	90
3	700	0.09	2	90
15	700	0.11	2	70
18	850	0.07	2	50
6	850	0.09	1	50
17	850	0.11	1	90
11	1000	0.07	1	70
1	1000	0.09	1.5	70
14	1000	0.11	1.5	50

The robustness of the data acquired through these experiments could have been affected and potentially improved by test repeats instead of wide ranges of parameters. However, the purpose of the research was mainly related to the objective of the final chapter and its machine learning algorithms. Therefore, it was deemed more relevant to widen the range of variable levels to improve data coverage for the interpolating approach of those algorithms, rather than testing the process variability and robustness through repeats. Ideally, both of these would have been beneficial, yet the time and budget constraints of the research limited this to one of the two options.

Along with the output of spark evolution data, tool wear values were also taken as outputs for wear analysis in the chapter 7. These last outputs were obtained as described in the beginning of this chapter, through a scale and a digital microscope, acquiring data of mass, crater wear and flank wear.

3.3.3.2 Image Acquisition

A video feed was recorded using a common SLR camera and the respective camera settings used can be seen in the Table 3.12.

Table 3.12: Camera settings.

PARAMETERS	VALUE
Resolution (pixels)	1280 x 720
Sample Rate (frames per second)	50
Aperture	5.6
ISO	400

The selected resolution and sample rate were the highest values that the camera could provide, whilst the aperture and ISO were chosen to partially obscure the image, as shown in Figure 3.16. This obscuring effect was found to improve the consistency of the recorded data, reducing the effects of random illumination, such as natural light and workshop lighting. It mainly aided in removing background noise, whilst maintaining spark information.

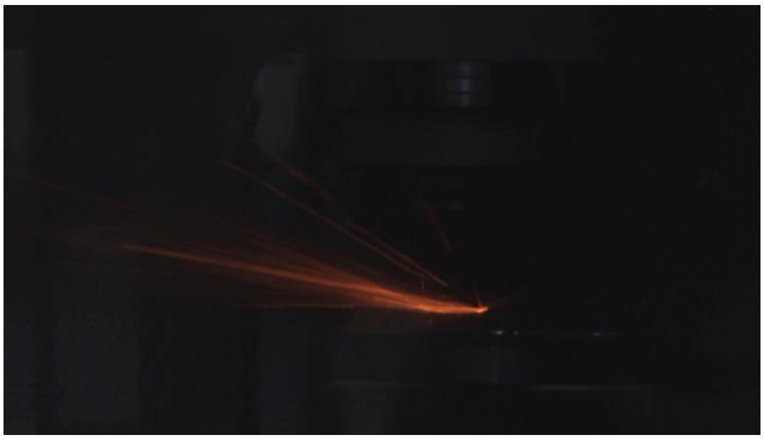


Figure 3.16: Partially obscured image using low ISO and wide aperture.

3.3.3.3 Machining and Parameters

The machining operation, as mentioned at the beginning of this section, had the same characteristics as in the second experiments (section 3.3.2.2). The workpiece was again an Inconel 718 round block and a square was machined for the tests, as shown previously in Figure 3.13 and Figure 3.15. Furthermore, the machining parameters used are described in section 3.3.3.1, and the Table 3.13 shows the calculated machining input. The real radial immersion had to be calculated for each combination of cutting depth and radial immersion percentage, as shown in the last column.

Table 3.13: Milling machining cutting parameters calculated from DOE.

TEST NUMBER	FEED SPEED Vf (mm/min)	SPINDLE SPEED n (rpm)	RADIAL IMMERSION ae (mm)
1	573	6366	32.2
2	401	4456	31.15
3	401	4456	42.3
4	595	5411	32.9
5	446	6366	42.3
6	487	5411	22.25
7	700	6366	40.05
8	312	4456	22.25
9	312	4456	41.4
10	490	4456	23
11	446	6366	31.15
12	379	5411	32.2
13	573	6366	23.5
14	700	6366	23
15	490	4456	32.9
16	487	5411	41.4
17	595	5411	40.05
18	379	5411	23.5

Every test included a total of 11 *runs*, up-milling the squared workpiece, and therefore delivering 11 videos and tool wear values for each test.

3.4 Summary

The general system configuration described at the beginning of this chapter was shared by the subsequent experiments, as well as the data processing analysis. However, as it was described by each experiment, the chosen devices, parameters and configurations where different depending on the specific objectives of that each experiment. Additionally, all the experiments were planned according to the different results and conclusions found in each experiment, constantly attempting to solve and improve the testing and processing conditions.

It could be seen how in the first experiments, a very simplistic system was used, with only still pictures as image acquisition feed, which presented limitations that had to be fixed, and this will be described in the chapter 5. Furthermore, these conclusions led to the introduction of a high-speed feed in the second experiments. Similarly, the conclusions drawn by these second experiments aided in the selection of the image acquisition device and the video feed chosen in the third experiments.

The purpose of including all the experiments here was to avoid repetition in subsequent chapters, as the data acquired from each test was used in each chapter to address different objectives. Therefore, during the rest of this written thesis, these experiments will be constantly referenced, and their relevant aspects will be revisited to analyse results and conclusions.

4 WEAR OF SIALON CERAMIC CUTTING TOOLS

The purpose of this research is to lay the foundations of a successful wear prediction system using image processing. In the literature review chapter, information regarding the wear mechanisms found in cutting tools was described. In the methodology chapter, on the other hand, the basis of the image processing techniques used to analyse and obtain data from cutting sparks was described. The current chapter has the twin aims of better understanding the tool wear mechanisms of SiAlON ceramic cutting tools and obtaining data that could be used in the later stages of the research. For this reason, the next sections will lay out the different wear assessment techniques used in this research to obtain wear data, as well as a comparison of other literature's findings with the current research's findings on tool wear assessment.

4.1 Wear Assessment Techniques

In the literature review some vision-based systems of direct TCM were reviewed, where most of them consisted of using images of cutting tools to obtain geometric measurements. This technique can be a useful in-situ procedure, with special effectiveness in measuring flank wear and crater wear.

To successfully compare the spark evolution of the machining processes with actual tool wear, it was necessary to obtain real tool wear values. As described in the methodology chapter, experiments included several runs of machining in each test, stopping the process between runs. This was done to obtain weight measurements and microscope images of the cutting tools. In this way, image processing could also be used to obtain wear data from the microscope images.

Therefore, this chapter includes the algorithms and procedures used to extract crater wear area and flank wear measurements from the microscope images. Furthermore, as an attempt to use image processing to find the volumetric loss of tool material, a stereo vision approach was investigated. The different results obtained with these techniques will be discussed, as the optimum techniques would be used for each experiment.

4.1.1 Flank wear and Crater Wear Area

The image processing algorithms previously created for the spark evolution had to perform spark analysis in a more "automated" fashion, given the number of frames or images for each test. Also, it was in the interest of the general research to build automated algorithms for a possible future integration in a stand-alone system. However, in the case of the cutting tools, a more supervised approach could be used, as the number of images per test was not very large.

For the extraction of flank wear measurements, a software named Tracker was used. This software enables the use of a calibration line tool to input a known geometric value, shown in blue in Figure 4.1. This calibration step uses these measured inputs to assign values to the image's pixels. For this reason, another software tool can be used to measure a distance accurately, shown in red. In this way, maximum VB_{max} and notch wear values could be extracted from the microscope images. Unfortunately, given that the insert lost material with the machining process, manual compensation had to be used to account for the uneven and worn tool edge.

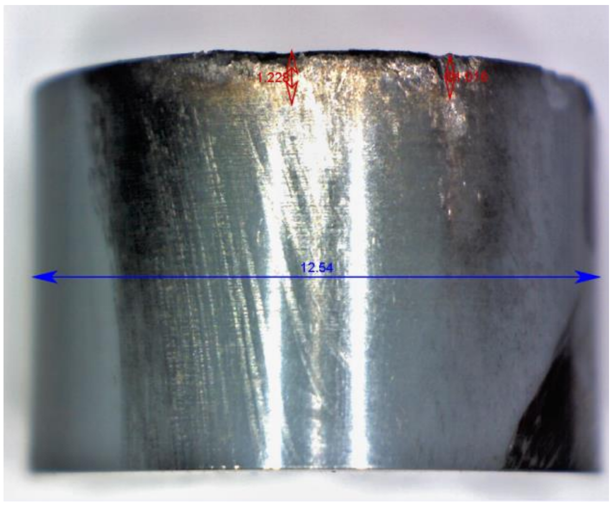


Figure 4.1: Flank and notch wear measuring using Tracker.

Whilst this seemed in the beginning as a simple and rapid task, it is relevant to mention that conducting manual measurement of each image ended up consuming much more time than expected. Therefore, it can be said that for future work a more careful recollection of images could have enabled the use of an image processing approach, like the one used next for crater wear area. Also, the measuring of each insert was deemed important for the implementation of the Neural Network, described further ahead in the chapter 7.

On the other hand, to extract crater wear area, an approach similar to the spark area extraction was taken, where an image processing algorithm was built to isolate, enhance and quantify the worn area. In Figure 4.2 (a), an image taken with a digital microscope of a SiAlON ceramic insert's rake face can be found. The wear area is easily distinguishable due to the change in colour and texture, compared to the rest of the insert and the background. Figure 4.2 (b) shows how the image was manually cropped to only process the area of interest. The next step consisted in extracting a section of the worn area to calculate means of the maximum values of each column. These values were later input into an image enhancement step shown in Figure 4.2 (c); the same enhancement step used in the spark processing algorithms, with the objective of isolating the worn area. After this, as shown in Figure 4.2 (d), different binarization thresholds were tested, selecting manually the best value that contained the least noise, as shown in Figure 4.2 (e). Finally, this binarized image is later filled, so that no empty spaces are found inside the worn area, as shown in Figure 4.2 (f).

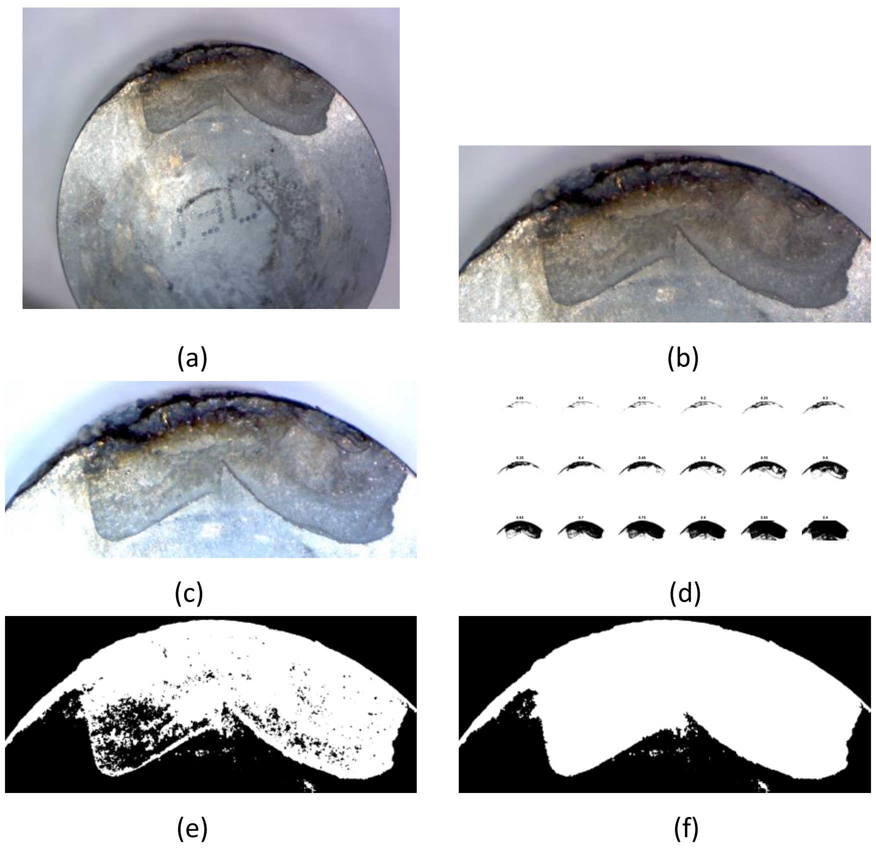


Figure 4.2: Crater wear area extraction: original image (a), cropped image (b), enhanced image (c), binary threshold selection (d), logical image (e), and final image (f).

Finally, after the worn area was successfully isolated, connected components discrimination and area computation was carried out (see section 3.2.1). The area extracted was in number of pixels, so an extra step was taken, where the algorithm is calibrated to know what is the size of each pixel in mm. This was done by counting the number of pixels that encompass the diameter of the cutting tool, knowing that the tool's diameter is 12.54 mm. Once this was calculated, the resulting value was squared to find the value of a pixel's area; this last value was then multiplied by the worn area in pixels to find out the actual worn area value in mm (algorithms can be found in Appendix II).

4.1.2 Stereo Vision

To assess wear as volume loss in the rake face of the ceramic insert, where most of the flaking and chipping occurred, it was decided to test a stereo vision, image processing approach. Stereo vision consists of processing two or more images of the same scene, matching specific pixels in these images. By doing this, the *disparities* of the analysed pixel positions can be used to calculate depth distances, and hence build a 3D representation of the scene (Szeliski 2010).

Figure 4.3 illustrates a stereo vision system of two cameras with the same focal length f, where Equation 4.1, Equation 4.2 and Equation 4.3 show how the value of disparity d can be calculated in such a system.

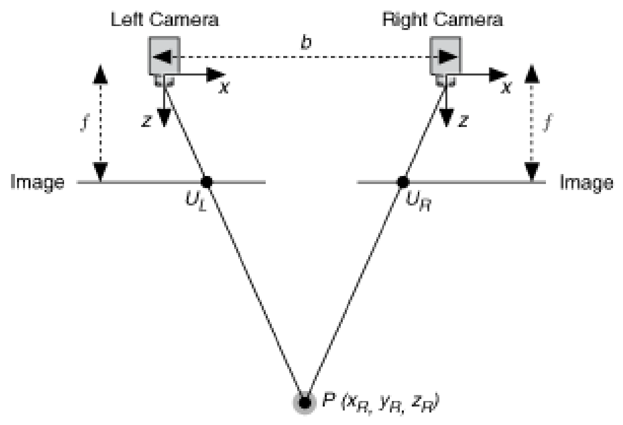


Figure 4.3: Stereo vision system using two cameras and equal focal length.

$$U_L = f\frac{X_{Left}}{Z_{Left}}$$

Equation 4.1

$$U_R = f\frac{X_{Right}}{Z_{Rright}}$$

Equation 4.2

$$d = (U_L - U_R) = f\frac{b}{Z_R}$$

Equation 4.3

Where U_L and U_R are the coordinates of the Real-world point P projected on the left and right camera sensors respectively; b is the distance between the cameras and Z_R is the Real-world distance to point P.

For the present research, the stereo vision system consisted of using an SLR camera (the same as the one used in the last set of experiments) with a LOREO 3D Macro lens for stereo vision mounted, shown in Figure 4.4. The acquired images would be processed using MATLAB's *Stereo Calibration App* for 3D reconstruction, with the objective of volume extraction.

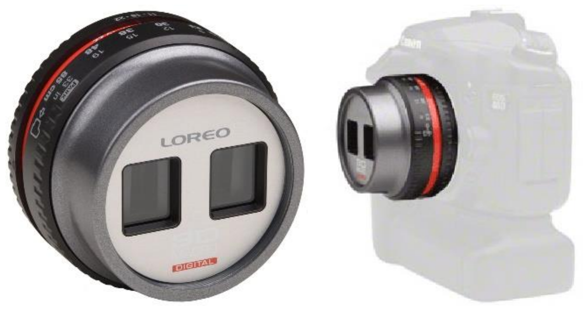


Figure 4.4: LOREO 3D Macro lens for stereo vision, mounted in Canon SLR camera.

The MATLAB's Stereo Calibration App calculates the intrinsic and extrinsic parameters of a stereo configuration of cameras for 3D reconstruction. To perform this, it was necessary to acquire and upload various images of a checkerboard in different orientations and locations. It was important to make sure that the square pattern was always visible, but that different cases of rotation and distortion could be included. Once applied, the calibration app detects points in the black and white square pattern, as shown in Figure 4.5. Through these points and by inputting the size of a checkerboard square in millimetres, the application calculated the disparities by computing their different locations about the rotations and distortions that were recorded. This system also provides both a graphic of reprojection errors, as shown in Figure 4.6, and a 3D visualization of the extrinsic parameters, as shown in Figure 4.7. The latter includes the orientation of the cameras (each sub-lens of the Loreo lens) in red and blue at the top right corner; as well the analysed checkerboards in different orientations and locations shown as squares in several colours.

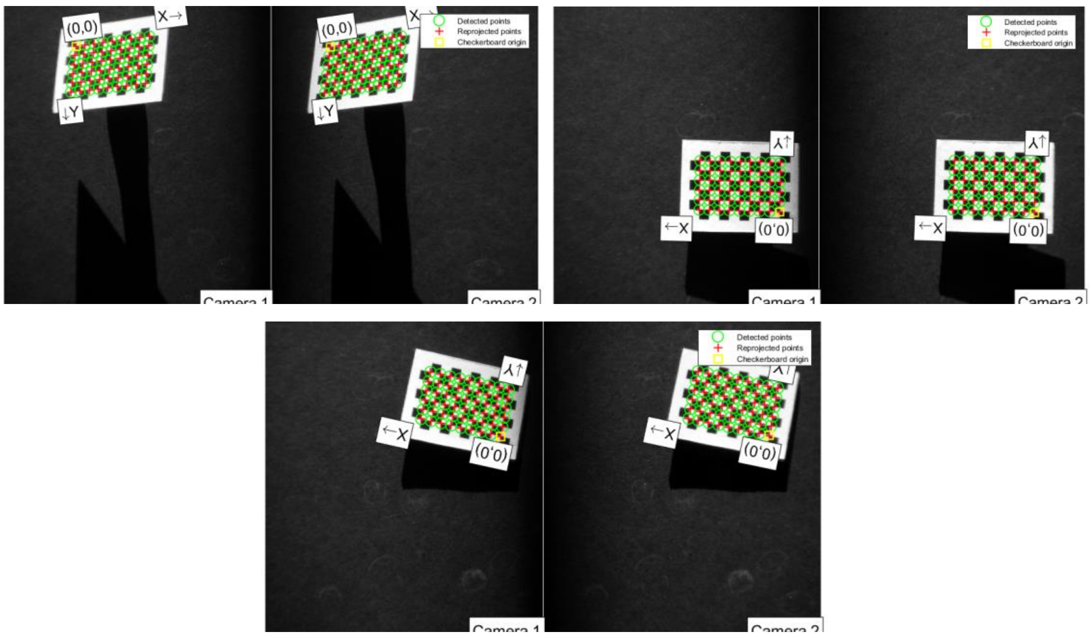


Figure 4.5: Examples of checkerboard points detection in different orientations.

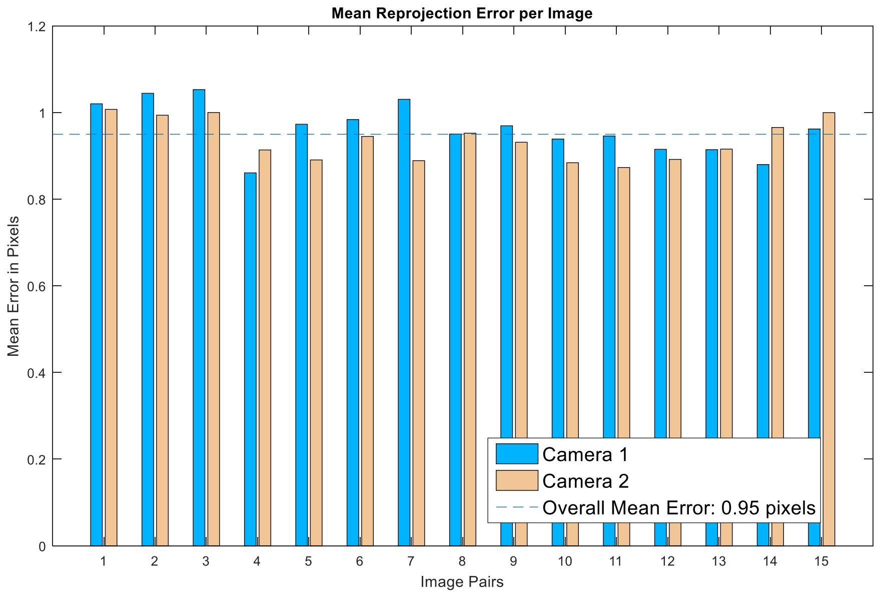


Figure 4.6: Stereo reprojection errors.

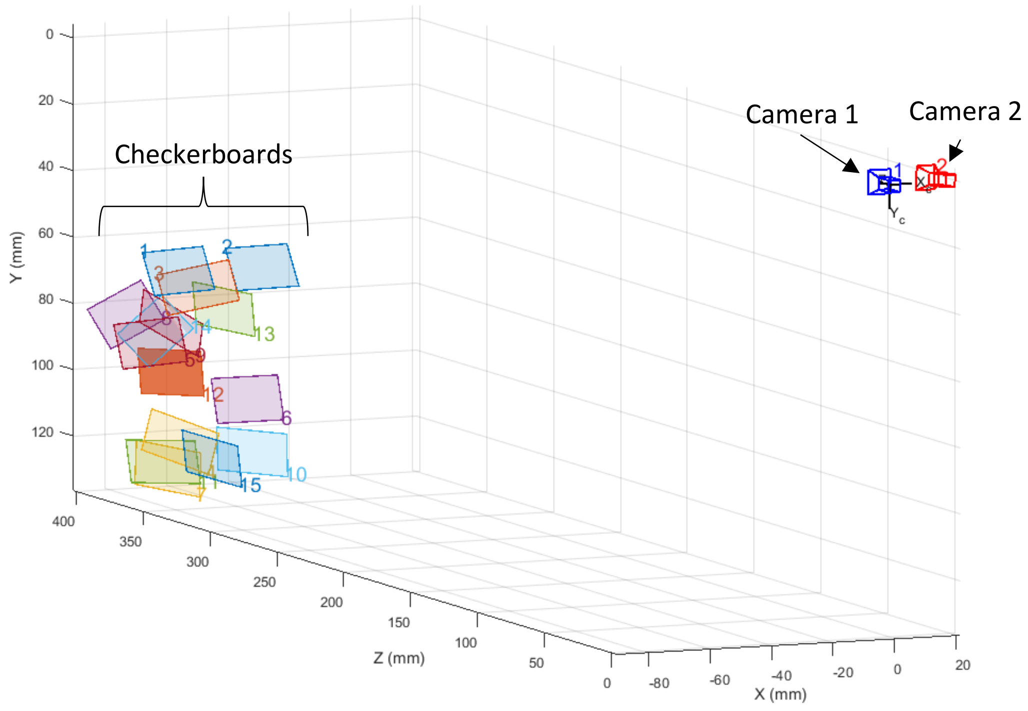
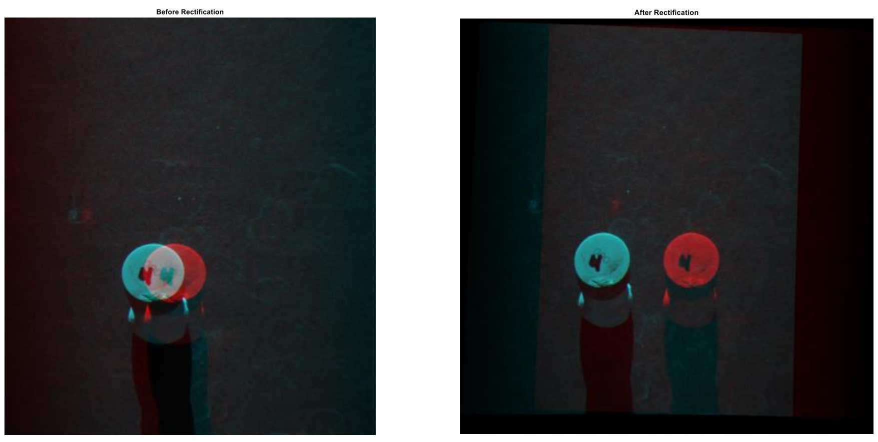


Figure 4.7: 3D visualization of stereo vision training system including analysed checkerboards (right squares) and cameras (top right).

Once these steps were taken, the stereo parameters that the application calculated were exported to the MATLAB workspace. A programme was then written to perform the 3D representation of a ceramic insert (Appendix III). Through this programme, a series of visual representations of the stereographic analysis of the insert were produced. Figure 4.8 (a) shows the images before rectification, and Figure 4.8 (b) shows the images after rectification.



(a)

(b)

Figure 4.8: Stereo images of insert: before rectification (a) and after rectification (b).

This rectification consisted of aligning each point in the left image, to its corresponding point in the right images, so that these were placed in the same row. This step is especially important to compute the new values of disparity of the insert's image. A visual representation of the disparity or distance between corresponding points can be found in Figure 4.9. In this figure, the pixels with stronger shades of red represent points with the highest disparity values whilst shades of blue represent lower disparity values or distance between pixels. It is expected that in a stereo configuration, objects closer to a lens should have higher disparity values, due to the perspective from each camera. Objects in the background, being comparatively "farther" have lower disparity values.

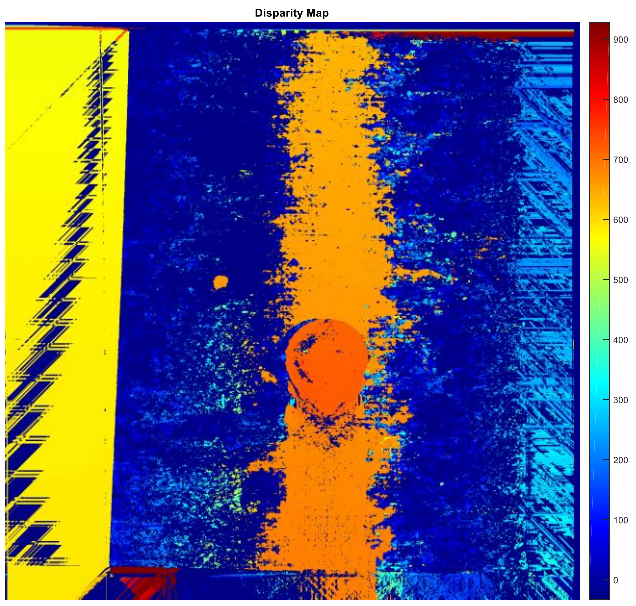


Figure 4.9: Disparity map of distances between pixels in insert's images.

The result of this stereographic analysis can be appreciated in Figure 4.10, where the 3D world coordinates of points corresponding to each pixel from the disparity map are used to reconstruct the scene.

It was found that the general geometric layout of the scene was successfully obtained by this approach. However, as it can be appreciated in the 3D scene, there is a high quantity of noise, yielding undefined shapes and lacking a good level of accuracy in volumetric measurements. Furthermore, the overall mean reprojection error value was of 0.95 (Figure 4.6), and although this was the lowest value obtained after several tests and configurations, it was still too high.

A “good” stereographic calibration would provide an overall mean reprojection error between 0.08 and 0.15 pixels. However, these values would be found in a scene where two cameras are used, and distances to objects could be between 1.5 and 4 meters. Therefore, it can be argued that in the present case, the configuration used could have contributed to this high error value, given that a macro lens was used, where the general focal distance was quite small, between 30 and 40 mm.

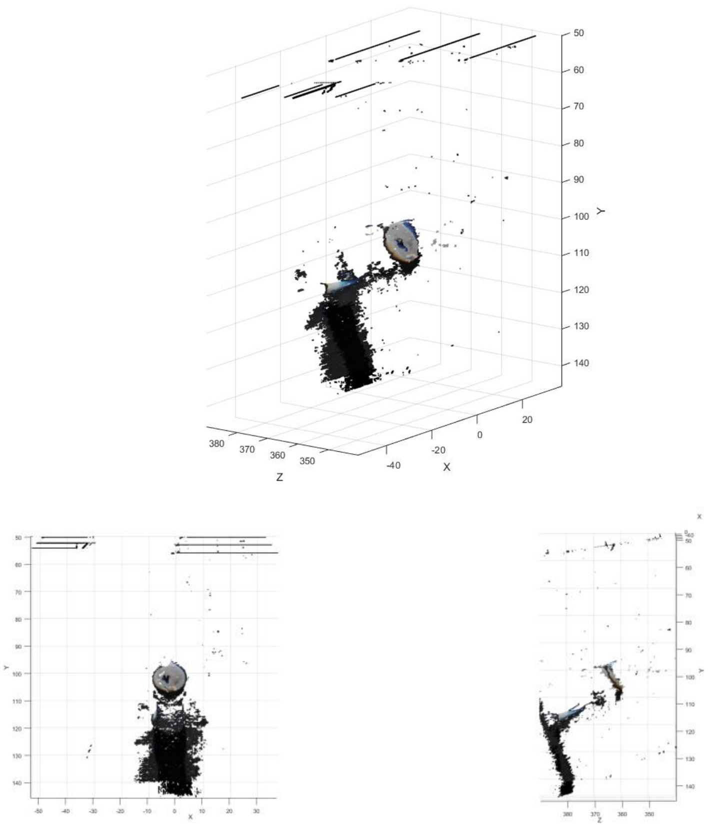


Figure 4.10: Views of 3D reconstruction of the insert from stereo vision analysis.

It was consequently concluded that this stereographic approach may not be an appropriate method to calculate the tool’s volumetric material loss, since the reconstructed scene was filled with noise, and for this application, high accuracy was necessary. 3D scanners in the market generally back up the stereo vision with some system of image correlation, where a map of points, lines or shapes are cast, so the calibration can be more accurate. Therefore, for future work, it could be recommended to explore the possible implementation of an image correlation system, along with stereo vision. However, in the author’s opinion, and with the results obtained, stereography alone does not seem to be accurate enough to become a successful tool wear assessment technique.

4.2 Tool Wear Assessment, Results and Discussion

In the literature review chapter, different properties and aspects of SiAlON cutting tools were explored, and a general overview of nickel-based super alloys was included. Furthermore, in section 2.1.3, a summary of the contributions of three authors who explored the same cutting tools and material used in this research was included. However, the present section will provide an extended summary of these references, comparing them with the wear mechanisms and parameters found during the current research experiments.

4.2.1 Literature Summary

In their work Xianhua Tian et al. (2013b) tested up and down milling with different high cutting speeds, and compared these with the wear present in SiAlON cutting tools, while also analysing cutting forces. They used a range of cutting speeds that varied from 600 m/min to 3000 m/min. It is important to mention that whilst this analysis of very high-speeds delivered diverse and interesting results, the manufacturer of the cutting tools (Sandvik Coromant) recommend the use of speeds between 700 and 1000 m/min. Tian et al. found the presence of the next secondary tool wear mechanisms:

- Crater wear (rake face wear) – Flaking and chipping of tool material occurred in the rake face, with a tendency to decrease with the increase of cutting speed (see Figure 4.11).

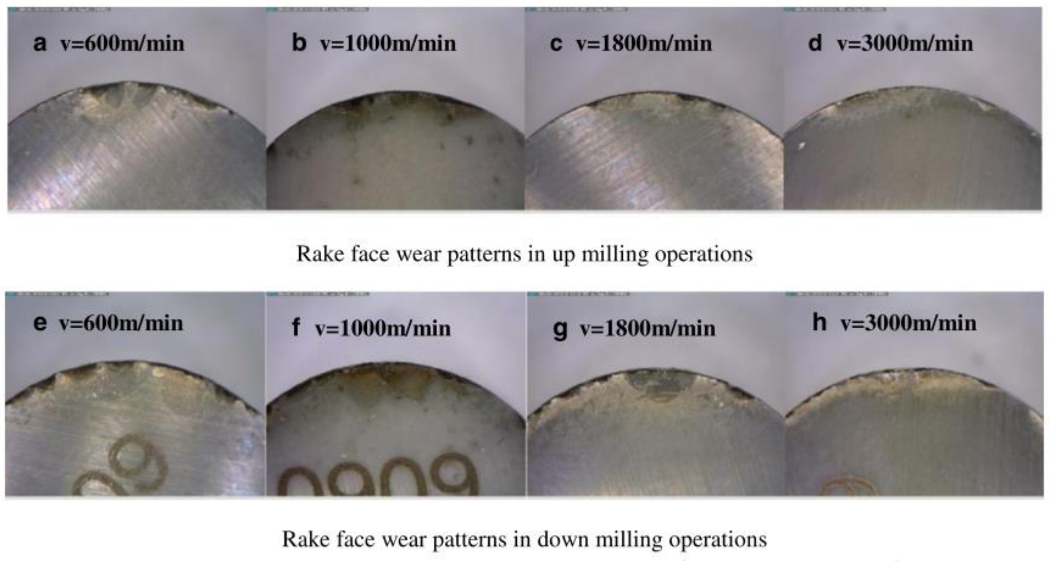


Figure 4.11: Rake face wear patterns under different cutting speeds (Xianhua Tian et al. 2013b).

- Flank wear – Dominant under higher speeds and in down-milling, edge chipping occurs due to the large compressive and shear stresses (see Figure 4.12).

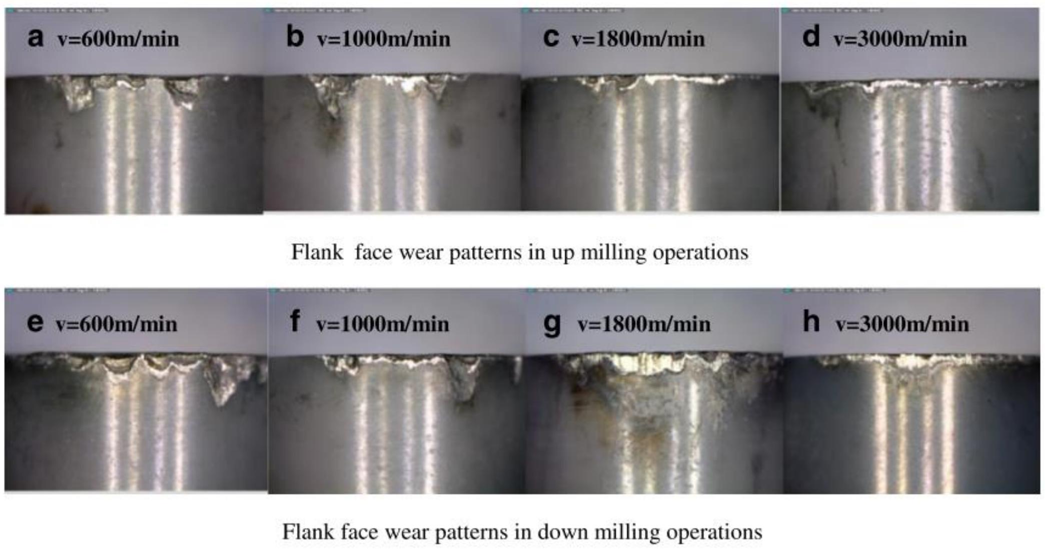


Figure 4.12: Flank face wear patterns under different cutting speeds (Xianhua Tian et al. 2013b).

- Notch wear – This was found to be very serious and dominant for tool failure at speeds up to 1400 m/min. Further increase of cutting speed presented a decrease of notch wear due to thermal softening.
- Microcracks – These were found on the flank face of the cutting tool, with an increase of cutting speed, and hence increase in temperature, these microcracks also increased in number (see Figure 4.13).

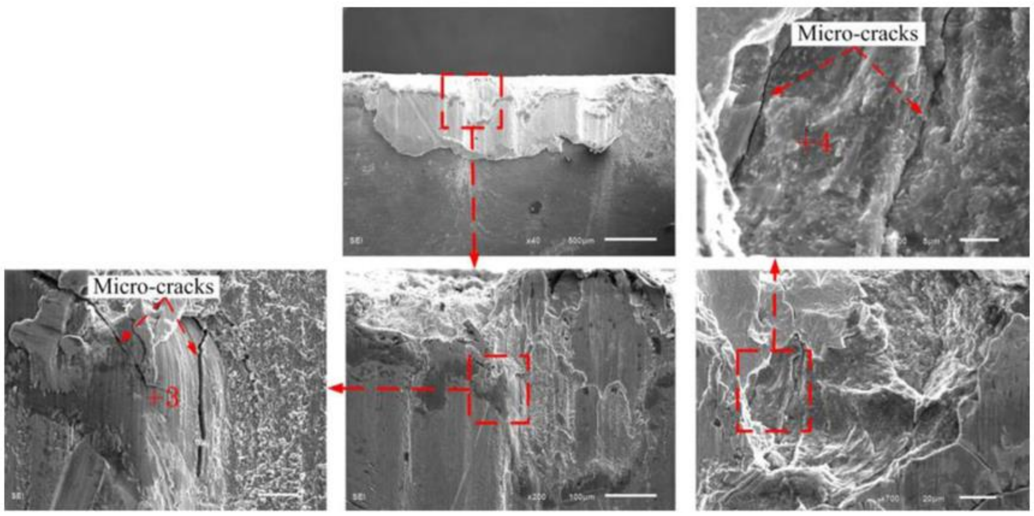


Figure 4.13: SEM images of the flank face at Vc = 1800 m/min (Xianhua Tian et al. 2013b).

They found a serious presence of adhesion when speeds ranged from 1800 and 3000 m/min, mainly due to the increase of temperature. Finally, they also discourage the use of cutting speeds under 1000 m/min when face milling, as notch wear becomes very serious and leading to tool failure.

Zheng et al. (2016) on the other hand, tested these materials and cutting tools using lower cutting speeds from 200 m/min to 1000 m/min. They also found similar secondary wear mechanisms and mechanisms, but the conclusions differed from the use of different cutting speeds. For instance, in the case of the crater wear, they also found flaking at speeds of 500 and 700 m/min, while adding the presence of adhesion after 900 m/min. For their flank wear analysis, they found this was more severe at lower speeds around 200 m/min, with some decrease with the increase of cutting speeds. As well, they found wear mechanisms of adhesion, abrasion and flaking in the flank edge of the cutting tool at 700 m/min. They also found evidence of notch wear, microcracking and chipping of the edge of the tool, with important crack propagation.

Finally, Renz et al. (2015) studied the tribochemical wear of SiAlONs at different sliding speeds on Inconel 718. They did not use Sandvik SiAlON cutting tools as with the other authors but rather they used a SiAlON block sample for sliding tests instead. However, they are the only authors that have analysed this pair of materials in a tribochemical approach. These authors divided the tool wear in “mechanically activated,” referring to the wear mechanisms explored by the previous authors, and “tribochemical wear.” They found through Scanning Electron Microscopes (SEM) and Energy Dispersive X-Ray (EDX) analysis, the presence of multiple “tribolayers” with mainly chromium and silicon oxides that, with the increase of sliding velocity and hence frictional power, could be divided into three phases:

1. Phase 1: at a low velocity of around 300 m/min and low frictional power, a brittle behaviour was found in tribolayer, increasing the COF and resulting in high wear.
2. Phase 2: higher speeds and an increase in frictional power created a stable and protective lubricious tribochemical layer, promoting optimal sliding conditions.
3. Phase 3: further increase in velocity and frictional power resulted in a dramatic increase in temperatures, where the tribolayer became mechanically unstable and started to breakdown and dilapidate. This, in turn, caused very high wear.

While this last author did not explore the mechanically activated tool wear, the speeds and temperatures they reported are in accordance with the other two authors. Furthermore, those two authors also used EDX analysis that matches the high concentrations of chromium, Silicon and Oxide, however, they used this information to support the presence of adhesion wear.

4.2.2 Tool Wear Assessment

Having performed an important number of experiments in this research, including different parameter and implementing different wear measurements (explained in section 4.1), it became relevant to make this research's own wear assessment. Even though wear measurements were carried out in all the three experiment sessions described in the methodology, the session that included a wide variety of parameters and results was the third experiment. For this reason, in this section, only the results obtained in the last set of experiments will be included and discussed. It is important to mention that the conclusions found, successfully matched the results and conclusions already found in the previous experiments.

Three main primary and secondary wear mechanisms were found in all the tests carried out. In some cases, certain primary or secondary wear mechanisms would be more intense than others, but in general, they all appeared at some point of the machining operation. The three main secondary tool wear mechanisms found were:

- Flank wear: this wear mechanism was found to be caused by the intense abrasive wear that occurred at the insert's flank face. This mechanism was present from the beginning of the machining process, originated by scar marks that were longitudinal and perpendicular to the insert's rake face. These scars increased gradually, intensified in some cases by chipping due to thermal wear and adhesion, as shown in Figure 4.14.
  Figure 4.15 displays a progression of five images (from within the eleven runs that each test included), showing how flank wear has a continuous and progressive behaviour.
- Notch wear: this wear mechanism tended to have a slow start, showing a small effect in the first runs. However, this mechanism dramatically increased well into the machining process, creating large scars and promoting chipping from thermal wear, as can be seen in Figure 4.14. In a few tests, notch wear was higher than flank wear, nevertheless, most of the tests showed larger flank wear than notch wear.
  Figure 4.15 also shows the progression of notch wear.

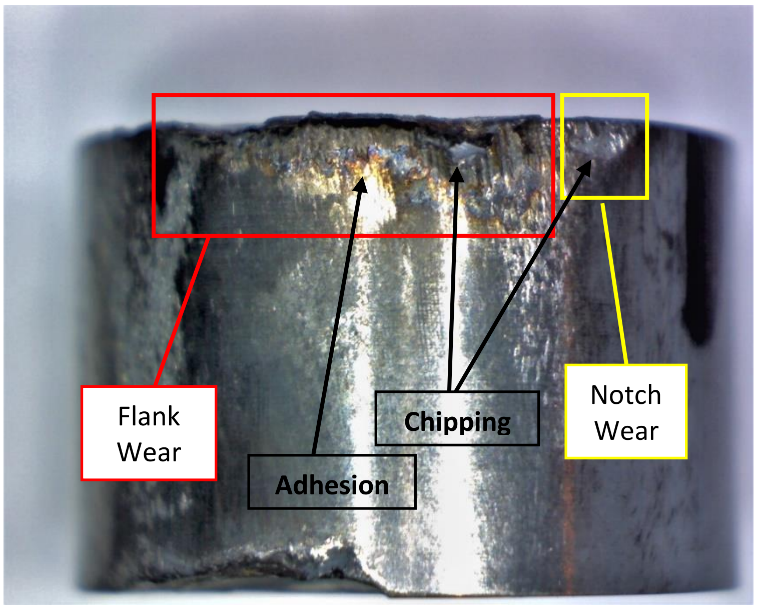


Figure 4.14: Primary and secondary wear mechanisms found in SiAlON inserts.

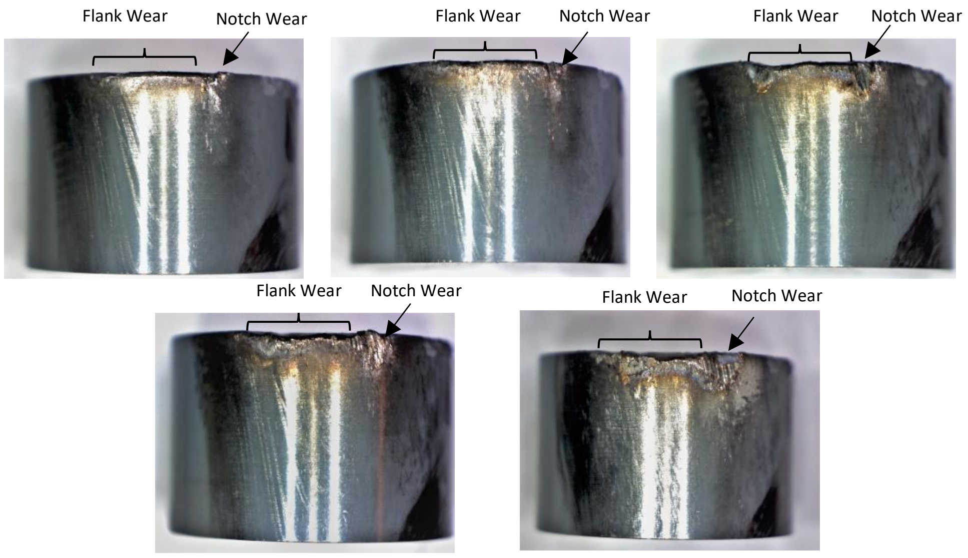


Figure 4.15: Sequence of insert's flank face microscope images in Test 6.

- Crater wear: this mechanism appeared from the beginning of the cutting process, gradually increasing in severity. It was found to be caused by the constant flaking and chipping of the inserts rake face, due to adhesion and thermal wear. However, this mechanism did not show a continuous or progressive behaviour, instead, crater wear seemed to increase by sudden leaps of severity. Figure 4.16 shows this behaviour through a progression of five images or stages of wear during a machining test. This wear mechanism was found to be a leading factor for tool failure, as this loss of material weakened the ceramic tool and increased the possibility of tool breakage.

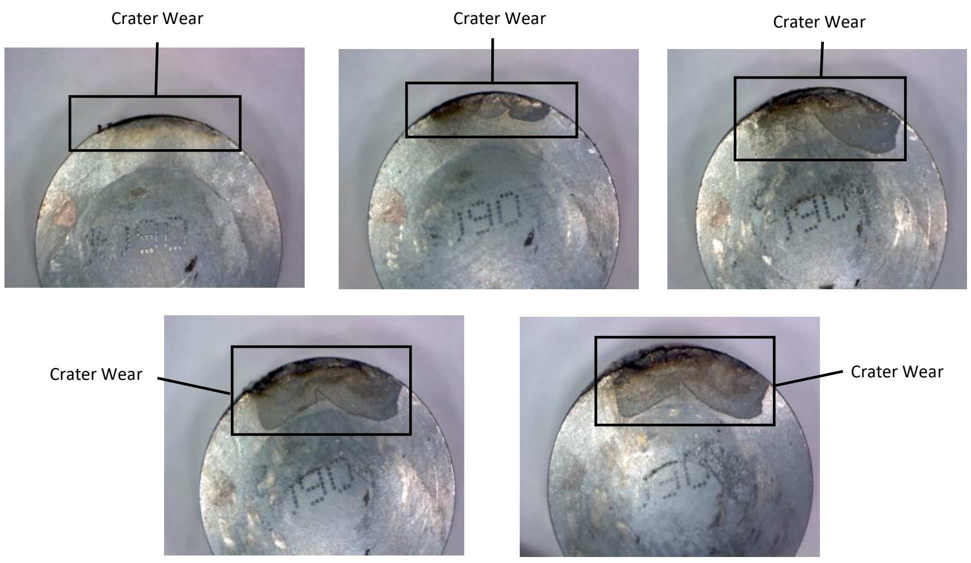
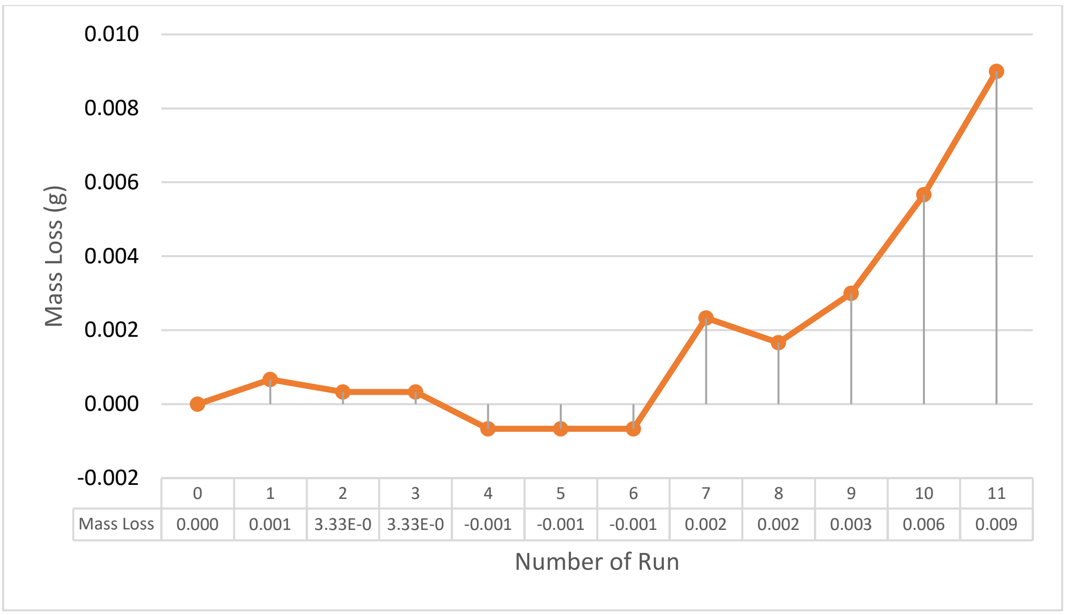


Figure 4.16: Sequence of insert's rake face microscope images in Test 3.

Regarding primary wear mechanisms, three main ones were found. Even though these were already mentioned inside the secondary mechanism descriptions, they were deemed relevant to analyse and explain individually; and these mechanisms were:

- Abrasion: this mechanism was evident and mainly present in flank wear, where the continuous sliding of the insert's flank face on the workpiece would promote high levels of friction. Due to the high temperatures and forces, the material is removed from the workpiece, as expected by cutting mechanics, however, particles from the cutting chips and breakage of the workpiece surface roughness would create long scarring of the cutting tool flank face.
- Adhesion: this mechanism can be found from early stages of the cutting process, identifiable by the incrustations of workpiece material found in the worn area after every run, as shown in Figure 4.14. Evidence of this can also be found in Figure 4.17, where the cutting insert's mass loss in Test 1 was obtained for each test run. From the 1st to the 2nd run, from the 3rd to the 4th, and even from the 7th to the 8th run mass loss decreased as the insert gained mass instead of losing it, and this is due to the adhesion of workpiece material.



	0	1	2	3	4	5	6	7	8	9	10	11
Mass Loss	0.000	0.001	3.33E-0	3.33E-0	-0.001	-0.001	-0.001	0.002	0.002	0.003	0.006	0.009

Figure 4.17: Mass loss in Test 1.

- Thermal wear: this last mechanism was expected in a process like this due to the high temperatures it involved, and it was present in all wear mechanisms. Inside the literature review in this subject, two authors talked about the development of microcracks as a cause of high cutting temperatures. Inside this research, it was not possible to carry out an analysis of these microcracks through SEM. However, it was mentioned by these authors, and confirmed by the different findings of this research, that these microcracks are present throughout the cutting process, developing flaking and chipping; both leading causes of tool dilapidation and failure

These primary and secondary wear mechanisms were found throughout all the different tests, however, they had different levels of severity. Therefore, the next section will explore the individual results obtained, to understand and discuss the relationship between wear and the machining parameters tested.

4.2.3 Results and Discussion

The individual results for each test of the third set of experiments can be found in the Appendix IV, however, Figure 4.18 shows the maximum values of flank wear, notch wear and crater wear. As may be appreciated, the values of these three wear mechanisms varied with each test. Figure 4.19 shows again the maximum values of flank wear and notch wear, along with the mass loss. In both figures, specific tests with high values of wear can be easily identifiable. However, there are different dimensions between wear descriptors and in both figures, the bar-type chart values (VBmax and VBN) are in accordance with the left vertical axis, whilst the area-type chart values (crater area and mass loss) are in accordance with the right vertical axis.

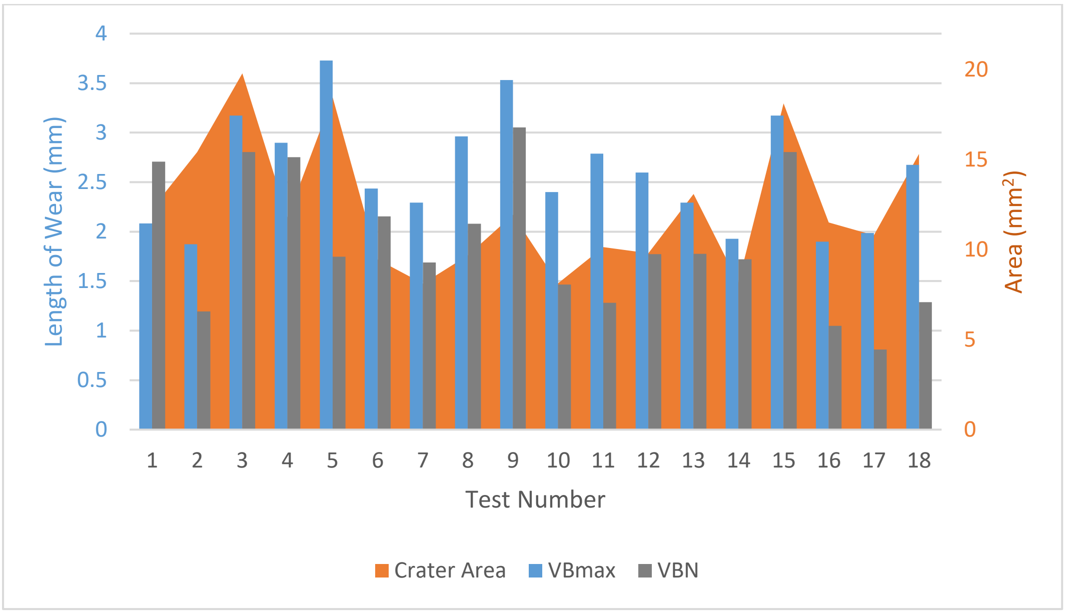


Figure 4.18: Maximum values of: Flank wear (VBmax), notch wear (VBN) (left vertical axis), and crater wear area (right vertical axis).

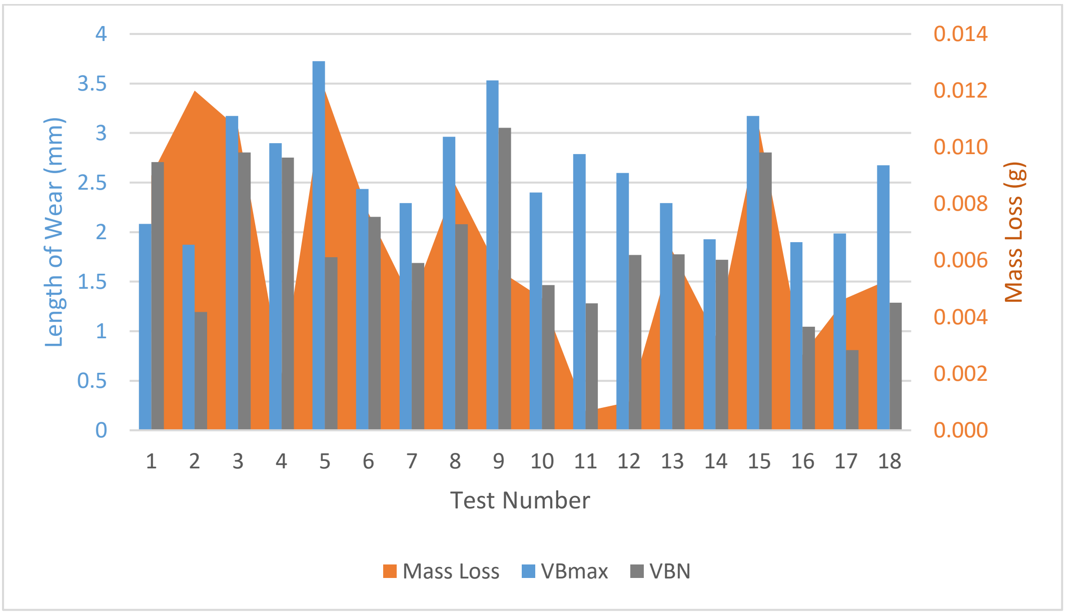


Figure 4.19: Maximum values of: Flank wear (VBmax), notch wear (VBN) (left vertical axis), and mass loss (right vertical axis).

As it was mentioned in chapter 3, the machining parameters of each test varied in accordance with a design of experiments. It is therefore deemed relevant to re-insert Table 3.9, to re-display the machining parameters for each test. This is shown in Table 4.1.

Table 4.1: Orthogonal array L18, generated using SPSS Statistics.

TEST NUMBER	CUTTING SPEED Vc (m/min)	FEED fz (mm)	CUTTING DEPTH ap (mm)	RADIAL IMMERSION (%)
1	1000	0.09	1.5	70
2	700	0.09	1	70
3	700	0.09	2	90
4	850	0.11	2	70
5	1000	0.07	2	90
6	850	0.09	1	50
7	1000	0.11	1	90
8	700	0.07	1	50
9	700	0.07	1.5	90
10	700	0.11	1.5	50
11	1000	0.07	1	70
12	850	0.07	1.5	70
13	1000	0.09	2	50
14	1000	0.11	1.5	50
15	700	0.11	2	70
16	850	0.09	1.5	90
17	850	0.11	1	90
18	850	0.07	2	50

In order to compare the different states of wear regardless of their dimensional variation, a scaling procedure was applied. The scaling was set between a range of 0 to 10 for each wear mechanism, where the maximum value was assigned the value of 10. This new dimension was named *wear severity* and Figure 4.20, Figure 4.21, Figure 4.22 and Figure 4.23 show all wear descriptors; grouped in the different machining parameters of cutting speed, feed, cutting depth and radial immersion.

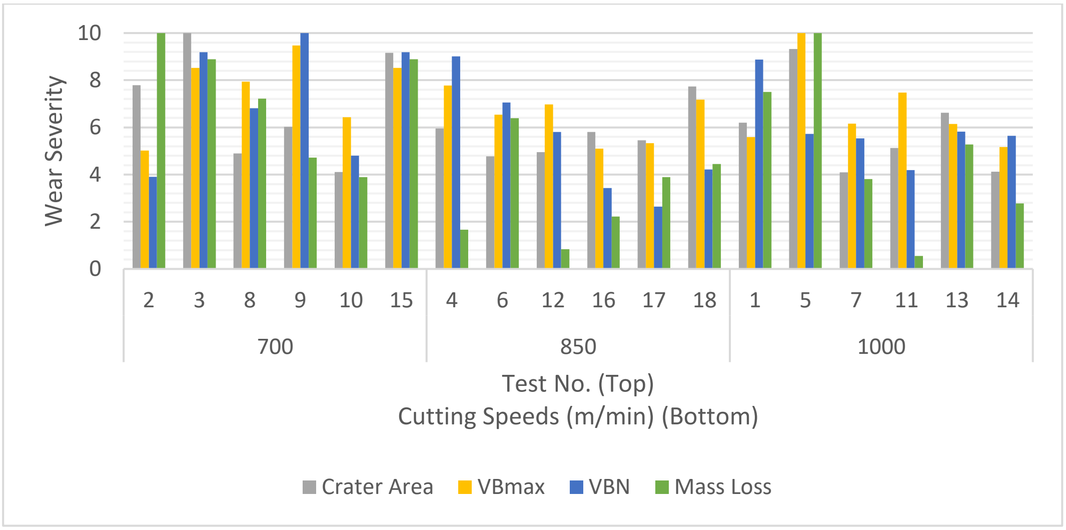


Figure 4.20: Wear severity of crater wear area, flank wear (VBmax), notch wear (VBN) and mass loss grouped in cutting speeds (Vc).

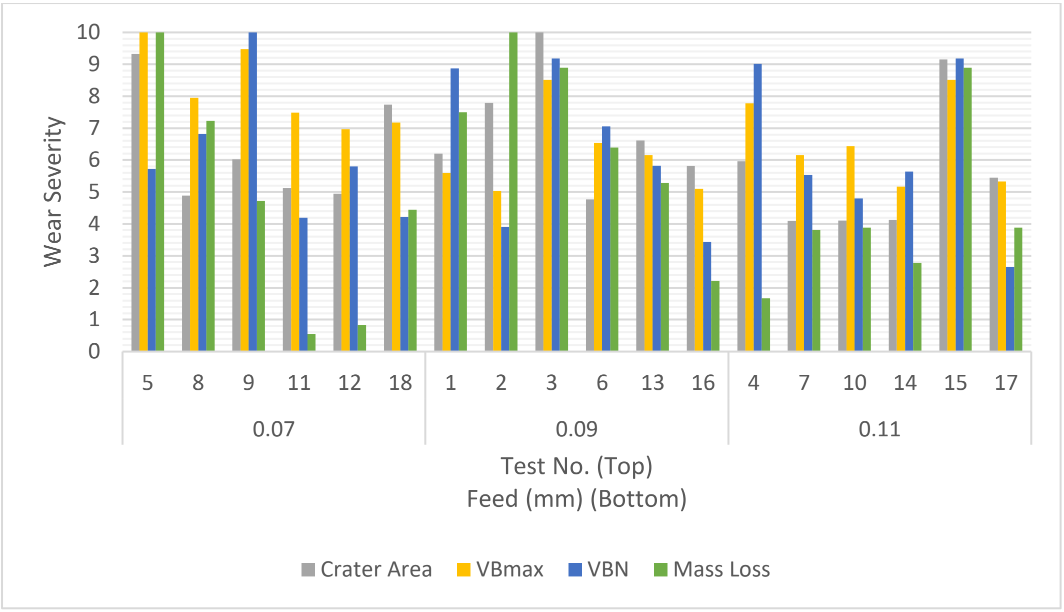


Figure 4.21: Wear severity of crater wear area, flank wear (VBmax), notch wear (VBN) and mass loss grouped in feeds (fz).

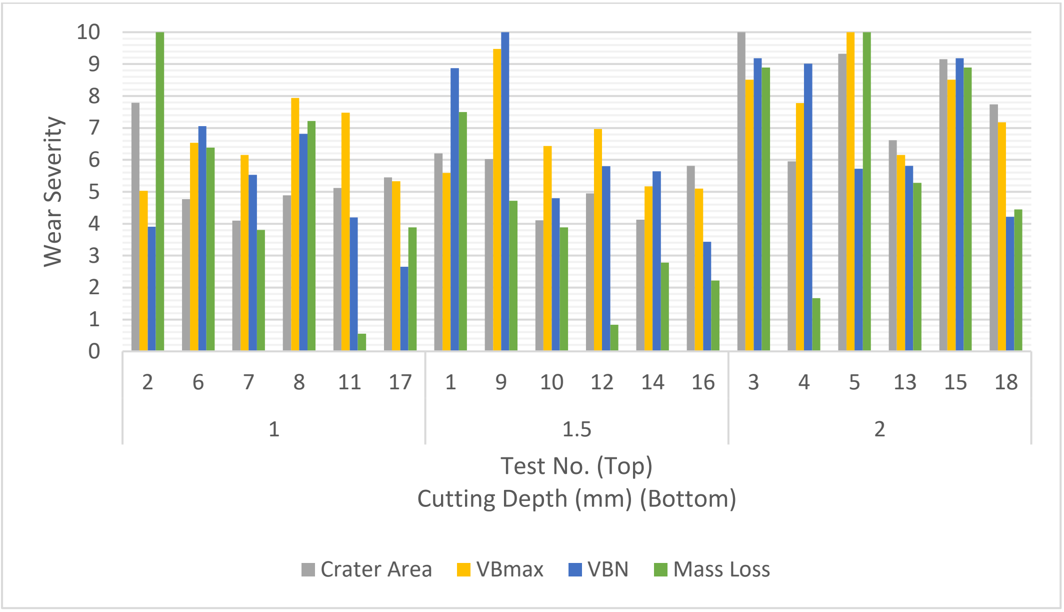


Figure 4.22: Wear severity of crater wear area, flank wear (VBmax), notch wear (VBN) and mass loss grouped in cutting depths (ap).

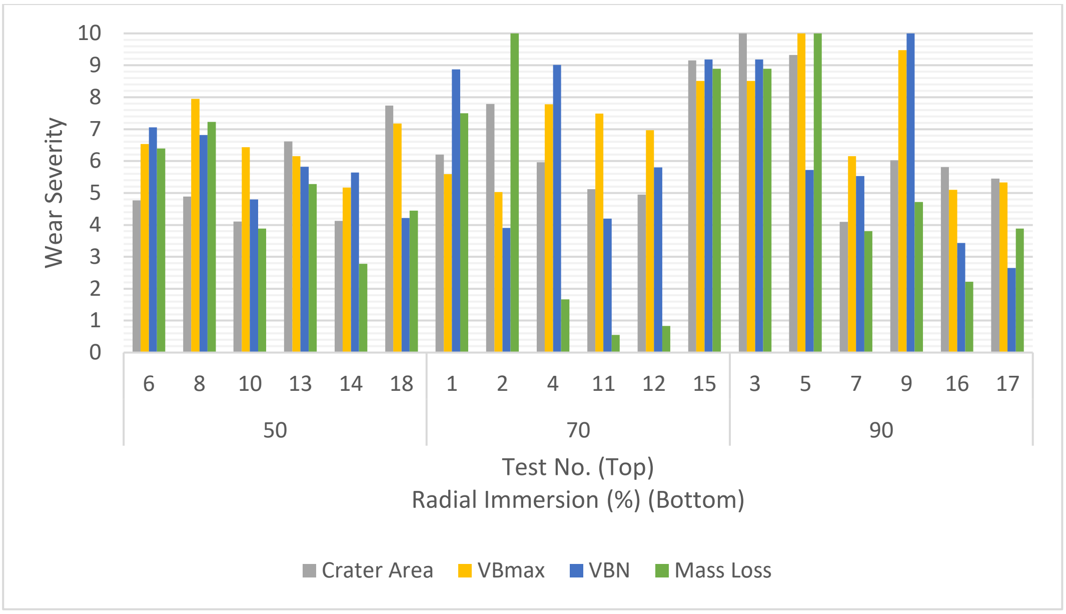


Figure 4.23: Wear severity of crater wear area, flank wear (VBmax), notch wear (VBN) and mass loss grouped in radial immersions.

As it can be appreciated in these figures, it is still difficult to identify relations or a dependency in a specific parameter. For this reason, a sum of wear severity values into a *total severity* value was carried out for each test and the results are shown in Figure 4.24, Figure 4.25, Figure 4.26 and Figure 4.27.

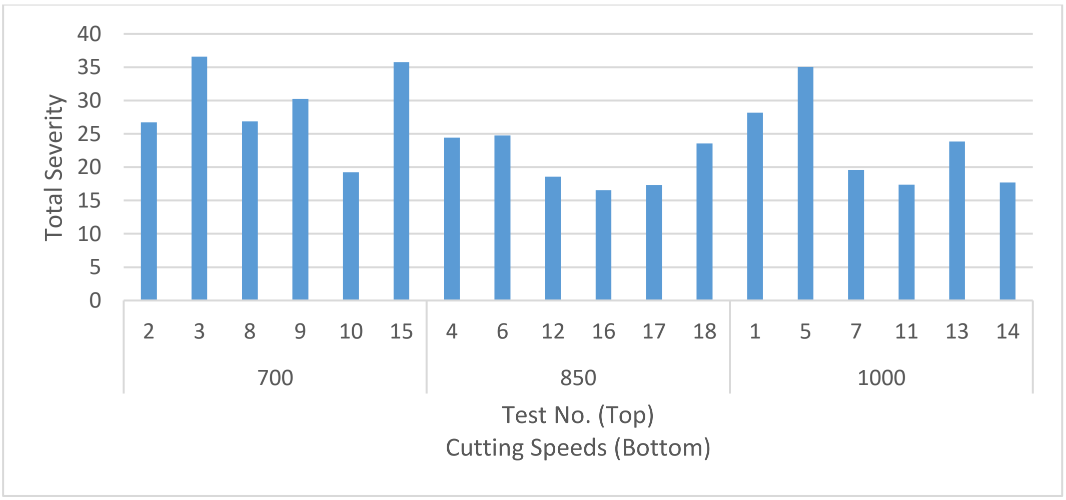


Figure 4.24: Total severity, grouped in cutting speeds (Vc).

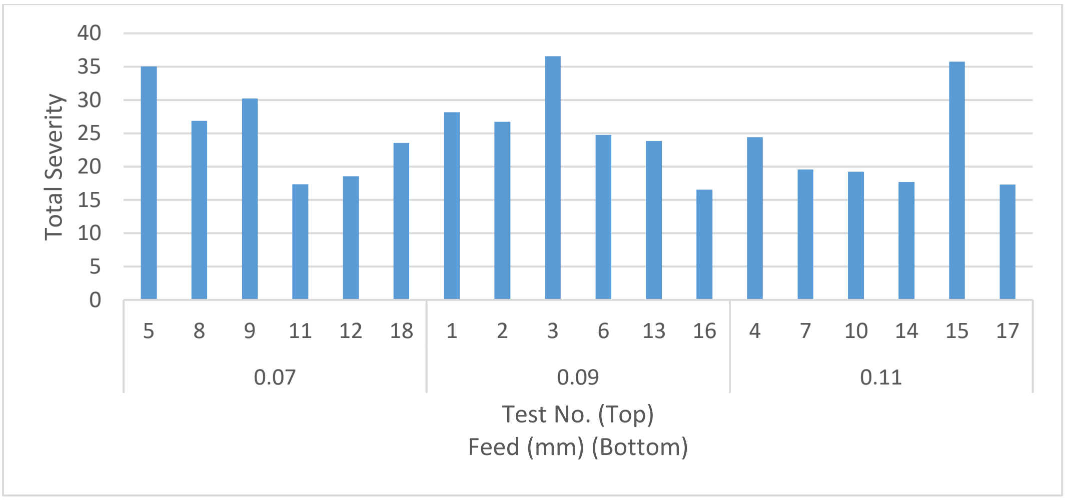


Figure 4.25: Total severity, grouped in feeds (fz).

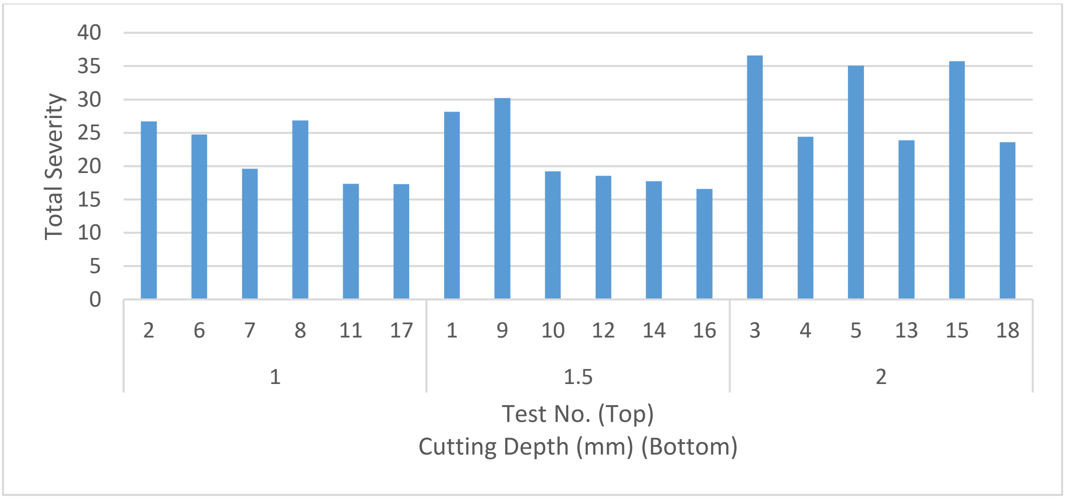


Figure 4.26: Total severity, grouped in cutting depths (ap).

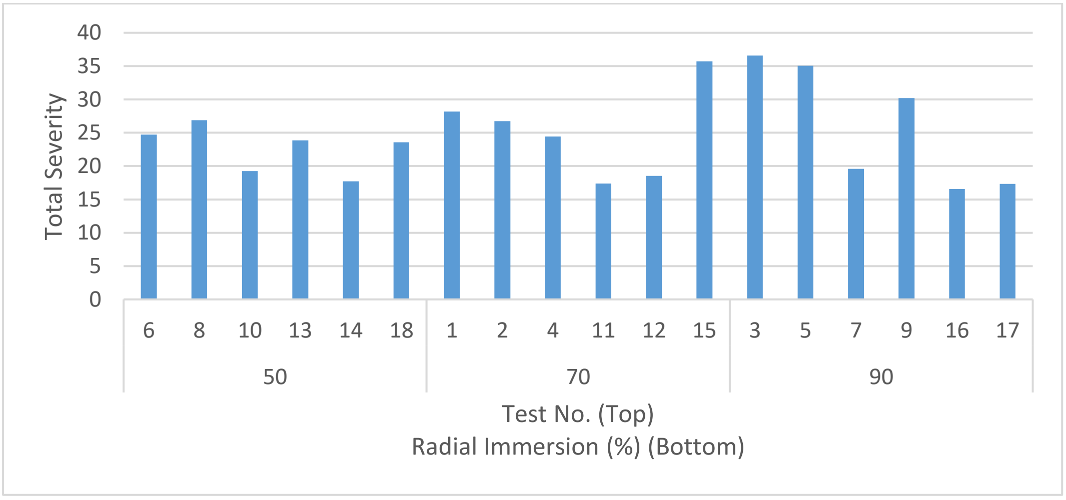


Figure 4.27: Total severity, grouped in radial immersions.

As it can be appreciated in Figure 4.25, the variation of feed did not seem to have a clear relation with the total severity of wear in the inserts, as the high and low values seemed to be scattered. In Figure 4.24 however, cutting speed variation seemed to suggest that the lowest values of total severity were found in the region between 850 m/min and 1000 m/min, nevertheless, this relation was not entirely clear. It was in the cutting depth and radial immersion, Figure 4.26 and Figure 4.27 respectively, that an increase in total severity was apparently found in the highest values of each parameter. Cutting depths of 2mm and radial immersions of 90% seemed to achieve very severe values of wear. Figure 4.28 shows this in a clearer manner by grouping the values of total severity, both by radial immersion and cutting depth. Again, this figure appears to indicate that more severe wear is likely to occur at higher values of radial immersion and cutting depth.

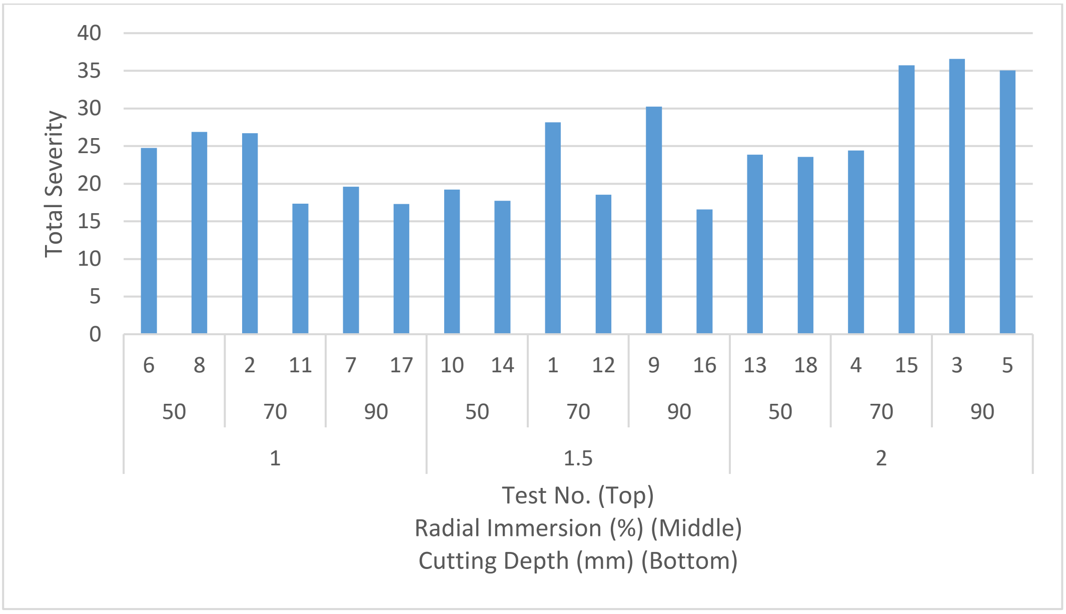


Figure 4.28: Total severity, grouped in radial immersions and cutting depths.

Nevertheless, and as shown throughout the figures, the results simply did not show clear trends or relationships. This could be related to the scatter that may be inside the actual trends. It could be argued that, with the methodology and experimental setup chosen for this assessment, it would be possible to mistakenly generalise the behaviours of these single tests' trends. Therefore, perhaps more tests or repeats could have aided in obtaining an acceptable level of statistical significance in the correlations mentioned before. However, it may also be the case that the standard deviation may be too high, showing that even with test repeats, these are not entirely representative. Regardless, as mentioned in section 3.3.3.1, the time and cost constraints of the project limited the amount of tests, but this could be a relevant future work.

Furthermore, Table 4.2 shows again the severity of each wear mechanism and total severity values on the far right. This table represents a qualitative approach to show the combined effect of all wear mechanisms, as the summation approach was arbitrary. Arguably, some weighting of these mechanisms or regression analysis could have been used to give a more quantitative reference of total wear severity. However, an in depth characterisation of the impact of tool wear mechanisms in tool failure, which would be required for the weighting or regression analysis approaches, was not deemed necessary for the research objectives and the research scope. Therefore, Table 4.2 simply attempts to show an overview of the wear mechanisms and their magnitude, as well as illustrate a simple comparative wear reference of each test.

A colour coding was applied to the total severity values to highlight the tests were the highest values were found. Tests 3, 4, 9 and 15 show the highest levels of severity, while tests 1, 2, 4, 6, 8, 13, and 18 show medium-high levels. The rest of the tests show mild to low levels of severity.

Table 4.2: Wear mechanism severity values and total severity.

Test No.	Crater Area	VBmax	VBN	Mass Loss	Sum of Severity
1	6.205	5.589	8.870	7.500	28.164
2	7.791	5.025	3.908	10.000	26.724
3	10.000	8.516	9.184	8.889	36.590
4	5.958	7.778	9.017	1.667	24.420
5	9.329	10.000	5.726	10.000	35.054
6	4.769	6.536	7.055	6.389	24.749
7	4.097	6.155	5.536	3.806	19.593
8	4.893	7.945	6.816	7.222	26.876
9	6.026	9.471	10.000	4.722	30.220
10	4.104	6.437	4.802	3.889	19.231
11	5.123	7.483	4.196	0.556	17.358
12	4.952	6.968	5.801	0.833	18.554
13	6.618	6.150	5.817	5.278	23.863
14	4.126	5.170	5.640	2.778	17.714
15	9.156	8.516	9.184	8.889	35.746
16	5.813	5.098	3.429	2.222	16.562
17	5.452	5.331	2.647	3.889	17.318
18	7.737	7.177	4.216	4.444	23.574

Except for test 10, the lowest levels were found in cutting speeds between 850 and 1000 m/min, and all of them included cutting depths between 1 and 1.5 mm. However, these lowest levels include the entire range of feeds and radial immersions.

When all these results are compared to literature, it is found that Tian et al. (2013b) suggested a decrease in crater wear with the increase of cutting speed, whilst this current research suggests that this wear mechanism is more dependent of cutting depth. Tian et al. (2013b) also suggested that flank wear was more dominant at higher speeds. However, the current findings are more in accordance with Zheng et al. (2016) suggesting that flank wear seems to be slightly more dominant at the lowest speed of 700 m/min.

Even though a Tribochemical analysis, similar to the one carried out by Renz et al. (2015) was not carried out, by analysing the different phases and the behaviour of the tribolayers that they described in their article, it can be argued that the results of the experiment have some correlation with their findings. They mentioned a series of phases that described the sliding contact between SiAlON ceramic and Inconel 718. These phases showed how there seemed to be an area of sliding parameters where the low values of wear can be achieved. The results obtained also seemed to suggest the existence of this area of low wear.

In general, it can be said that a “linear-like” behaviour, such as it is suggested by both authors in their articles, was not evident with the variation of machining parameter. Instead, as mentioned in the previous paragraph, the findings of this research seem to suggest that there is a window of optimum parameters to assure low values or wear. This, in turn, can be attributed to the brittleness of ceramic materials, which seemed to initially create randomised values of wear. Yet, as it was described before, some relations were found through the different tests carried out. Nevertheless, it can be argued that the number of tests included in the design of experiments, along with the selected values of variables and levels, may not be enough to carry out a calculation of this window of parameters. Furthermore, since the scope of this research does not encompass the optimization or characterisation of these cutting tools’ parameters, this window will not be suggested. However, this could be an interesting topic for future work.

4.2.4 Summary

As not many studies on SiAlON tool behaviour were found, this chapter attempted to enrich this area by filling some gaps. Therefore, the two main objectives of this chapter were: (1) to describe the different techniques explored in the present research to measure wear, and (2) to perform a tool wear assessment to identify the main wear mechanisms involved.

The approaches described for tool wear measurement included a basic geometric measurement of wear through image processing, and an approach to tool wear volume measurement through stereo vision. It was found that the first approach was successful in generating wear data, with high efficiency in the crater wear area measurement, but some inefficiency regarding man hours in the flank and notch wear measuring approach. On the other hand, the stereo vision attempt to measure wear volume was unsuccessful, mainly due to the low level of accuracy that this approach produced. Therefore, the techniques of geometric measurement through image processing of flank and notch wear, as well as crater wear area, will be used in the subsequent chapters.

Regarding tool wear assessment, this chapter found four main wear mechanisms worth exploring; these were: crater wear area, flank wear, notch wear and mass loss. These findings were quite relevant for the main objectives of this research, and therefore the analysis of crater wear would be used to explore the research hypothesis in the next chapter. Afterwards, in the chapter 7, some of these wear mechanisms will be used for the prediction of tool wear.

5 IMAGE ACQUISITION SYSTEMS AND PARAMETERS

Throughout the current research, improvements were made to methodological aspects of the experiments and to the data processing algorithms. One of these improvements was the selection of the image acquisition systems and parameters implemented in each experiment. This is generally an important aspect of any analytic vision-based system, where the procurement of successful and accurate data is essential, and this is almost entirely dependent on the hardware used and their respective settings. It was, therefore, decided to first compare the results obtained in the first set of experiments, which included the use an SLR camera with different shutter speeds. After this, and due to some conclusions that were drawn, the second set of experiments were conducted using a high-speed video camera.

In general, the main objective of the present chapter was to present the process of selecting the optimal image acquisition system and parameters for a vision-based TCM system of spark-wear relationship. Additionally, it is important to mention that all this chapter's results and conclusions were published, reviewed and presented as a conference article by the author (Dominguez Caballero et al. 2016a).

5.1 Methodology

The first and second set of experiments were used to evaluate the best image acquisition system and parameters in this chapter. Therefore, as a summary of the first set of experiments described in the section 3.3.1, these consisted of four tests using fast, medium and slow shutter speeds respectively. However, as it was mentioned then, only the first three tests (test 1, 2 and 3) were used for the present study. Furthermore, images in all these tests were acquired at a constant rate of 1 image every 5 seconds throughout the machining operation.

In the second set of experiments described in the section 3.3.2, on the other hand, two tests were carried out, one included 8 runs and the other 5 runs. This difference was mainly due to the variation in machining parameters, the amount of workpiece material and the state of wear of the insert at the end of the tests. Furthermore, only one insert was loaded into the tool holder to isolate the wear values. Also, a different machine and workpiece material to that used in the previous experiments were introduced. This change of materials and machine was due to availability, however, these variations were not believed to affect the present study, as only image acquisition systems and parameters were analysed. Furthermore, given the conclusions that were drawn in the first set of experiments, a new image acquisition system was introduced. A Phantom v210 high-speed camera was used, recording a video feed instead of still pictures, at a high frame rate of 300 fps.

5.2 Still Pictures Using SLR Camera and Parameters

In an SLR camera, the amount of light captured by the imaging sensor is controlled by the shutter speed, the diaphragm (or aperture) and the ISO value (Nakamura 2005). This makes these three variables dependent upon each other, as varying the shutter speed would mean varying the other two as well. Therefore, to test different shutter speeds, it was decided to leave the ISO value as constant and simply adjust the aperture to enable the selected shutter speed. For this reason, the three speeds tested, included previously in the description of these experiments were of 1/640 (fast), 1/125 (medium) and 1/5 (slow) seconds.

Figure 5.1 shows the resulting images obtained using each one of the shutter speeds. It can be seen that, for fast settings (a), the spark seemed quite sharp but segmented. This was problematic at the processing stage of the spark features, as the randomness of the spark delivered very scattered results. The medium settings (b) on the other hand, provided a more solid spark, however, there was still some segmentation and randomness of the spark outline. Slow settings (c), successfully managed to acquire a solid and well-defined spark, with a clear outline and structure that would enhance the readability of the spark evolution results.

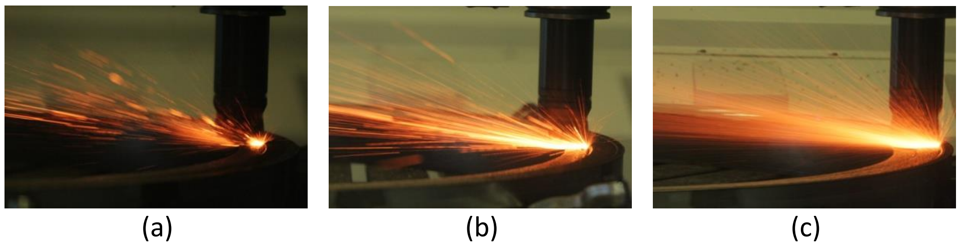
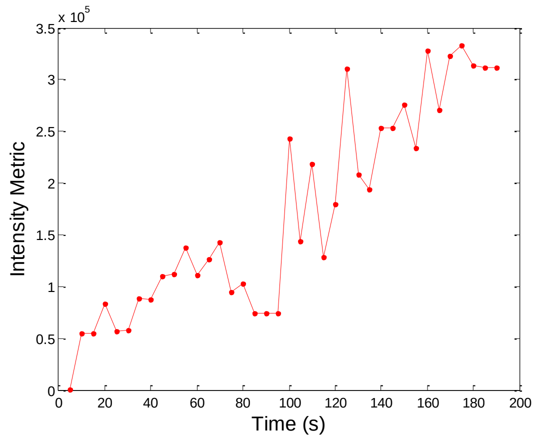
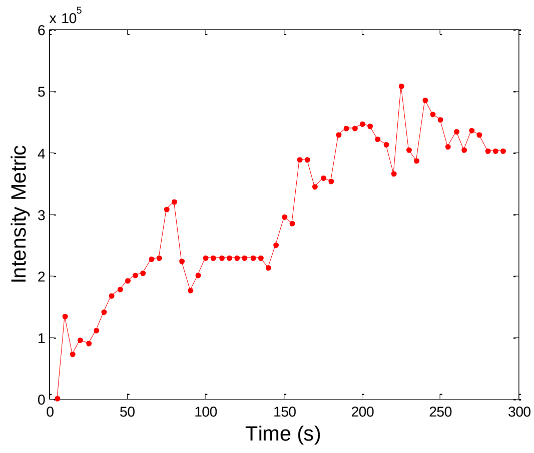


Figure 5.1: Resulting images of first experiments' tests using fast (a), medium (b) and slow camera settings (c).

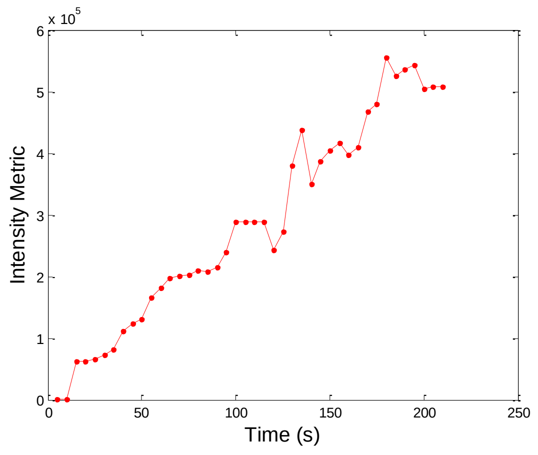
The next step was to extract the spark descriptors for each set of speeds, to compare the readability and behaviour of each set of settings. Nevertheless, when implementing the corresponding algorithms, the already mentioned randomness and segmentation of the sparks when using fast and medium speeds yielded unreliable spark area results. For this reason, Figure 5.2 only shows the spark intensity evolution for each test and speed. In this figure, it can be seen how the fast settings (a) produced the expected scattered and spiky results that would be unreliable, as some jumps of intensity could be caused by the spark randomness, instead of its actual growth. The medium speed results (b) produced a smoother evolution, but still showing sudden jumps that have the appearance of noise, again due to the spark segmentation and randomness. The slow settings (c) however, as expected, produced a much smoother and semi-continuous spark evolution.



(a)



(b)



(c)

Figure 5.2: Spark Intensity in first experiments for tests 1: fast settings (a), test 2: medium settings (b) and test 3: slow settings (c).

Given that, in this first set of experiments, the biggest challenge of the spark algorithms was to successfully isolate the spark from the background, the mentioned segmentation and randomness caused by medium and fast speeds, increased this difficulty. On some occasions, the spark isolation algorithms would choose the wrong section of the spark or a strong glass reflection, creating those random variations of the spark evolution trend.

Furthermore, there was an important limitation related to this type of image acquisition system. The sample rate of these tests was significantly wide, and there was plenty of spark information was lost in the 5-second spaces. For this reason, the conclusion of a more favourable set of results through slow settings was carried on, however, another image acquisition system with more robustness seemed necessary to test this statement further.

5.3 Video Feed Using High-speed Camera and Parameters

The conclusions mentioned above led to this second set of experiments, where a high-speed camera, recording a video feed was chosen. The wide gaps of data in the first experiments were solved by the introduction of a video feed, which is a near-continuous way of acquiring image data. Also, the need for a more robust system was addressed by using a high-speed feed. The idea was to record the high-speed video feed in a high frame rate, so that again fast and slow acquisition setting could be compared with a single set of data. This also enabled the extraction of as much information as possible.

The camera manufacturer's software enabled the segmentation of the high-speed video in images of each video frame, therefore delivering 300 images per second. However, given that a high-speed camera and only one insert were used, the process was quite intermittent and some images had almost no visible spark, as can be seen in Figure 5.3.

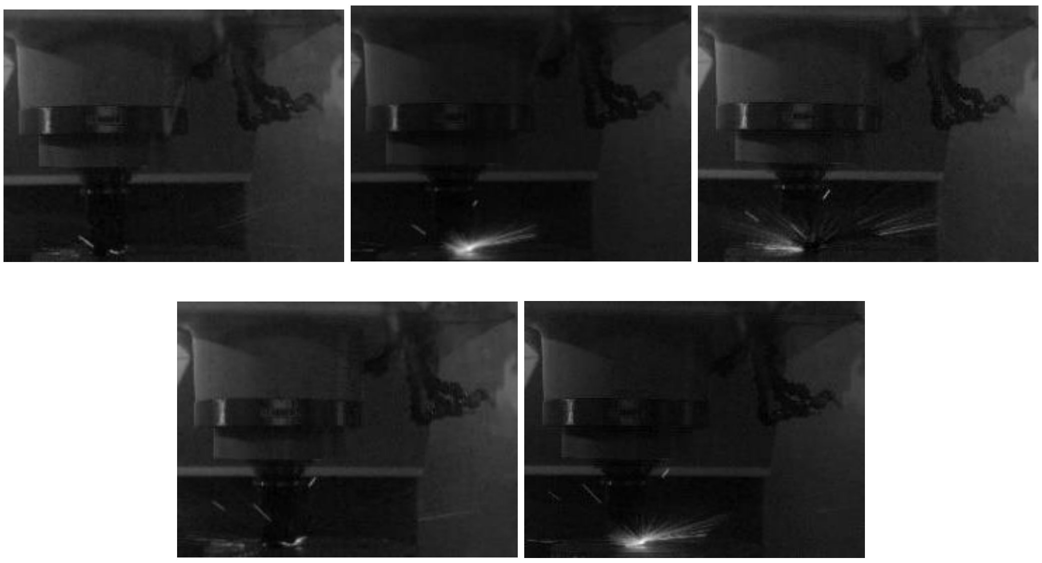
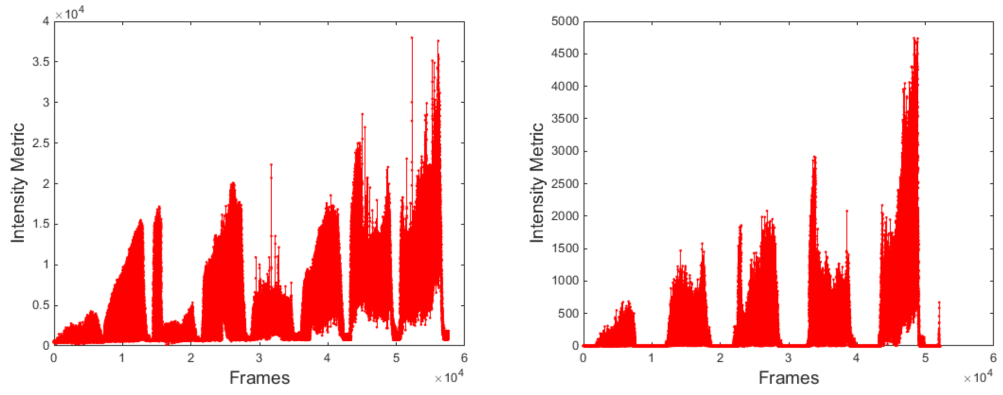


Figure 5.3: Second experiments, Test 1 - Run 4: sequence of frames 2356, 2357, 2358, 2359 and 2360 respectively.

Therefore, the use of the area extraction algorithm was again not possible, as this algorithm would try to find bright areas like the spark, extracting the area of the whole image instead. For this reason, it was decided to again only use the intensity algorithm to produce intensity evolution graphs from both tests, as shown in Figure 5.4. These graphs show how intermittent and noisy the extracted intensity signal was once processed.



(a) (b)

Figure 5.4: Second experiments spark intensity evolution for Test 1 (a) and Test 2 (b).

To smooth this noisy spark intensity trend, some sort of filtering technique was required, and between the many options available, a moving average was selected. Moving averages are commonly used to smooth quick variations of nearby points, which is the case of the spark intensity trend, and the code used for this filter followed Equation 5.1.

$$y_{average}(n) = \frac{1}{W}\Big(y(n) + y(n-1) + \cdots + y\big(n-(W-1)\big)\Big)$$

Equation 5.1

Where $y_{average}$ is the new averaged value of y data points inside the window W.

To select the moving average window size W, kurtosis values were extracted using different window sizes, as shown in Figure 5.5. Kurtosis is the measurement of the "peakedness" of a distribution, and it can help selecting a window size value that smooth data while keeping important information.

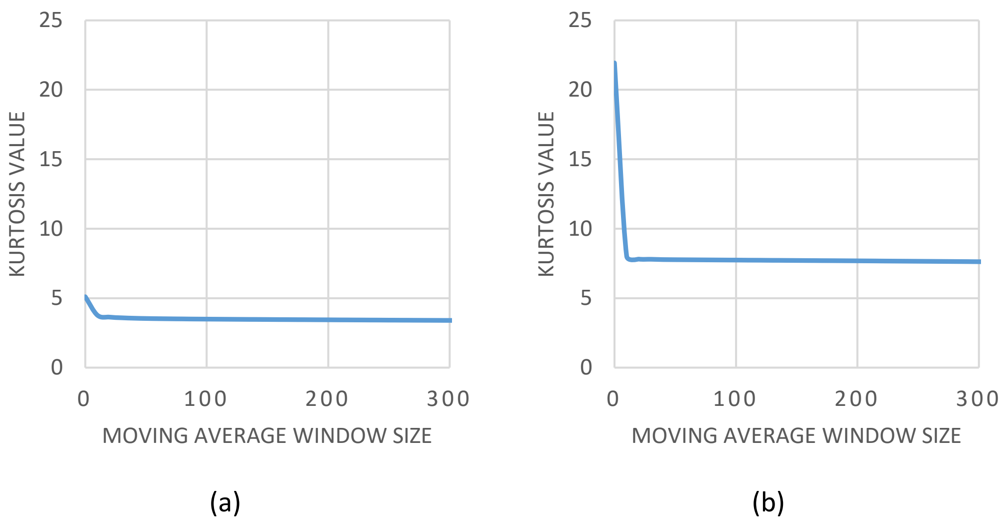


Figure 5.5: Kurtosis values for Test 1 (a) and Test 2 (b).

It was found that in a range between 10 and 600, some very similar overall smoothing results were perceived in the intensity graph, and even though kurtosis settled down at around 50, in closer inspection some variability was found. In the range of 300 however, there was a good balance between extracting relevant data and obtaining a general smoothness in the trend's outline. In Figure 5.6 the result of the implementation of the moving averaging filter can be found, where the intensity averaged values produced the much readable outline of spark intensity. This made in turn, easier to analyse and understand the spark behaviour, and it became clear once again that very instantaneous images were too problematic.

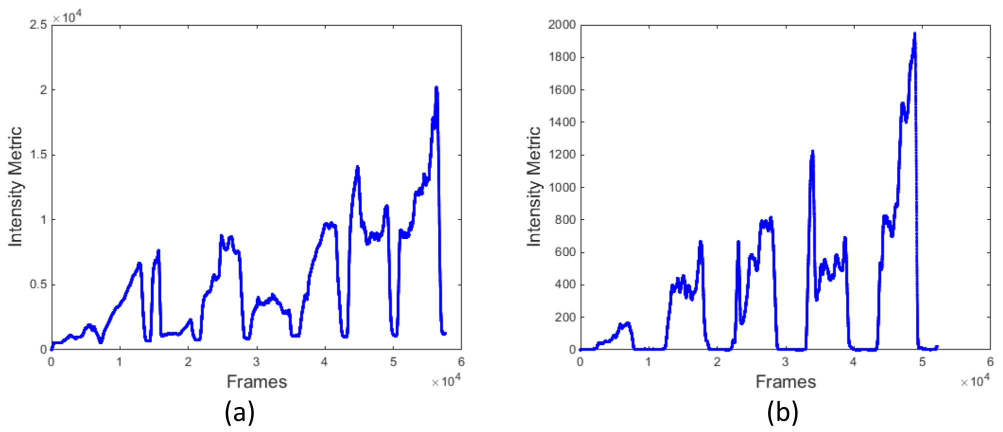


Figure 5.6: Averaged Spark intensity evolution for Test 1 (a) and Test 2 (b).

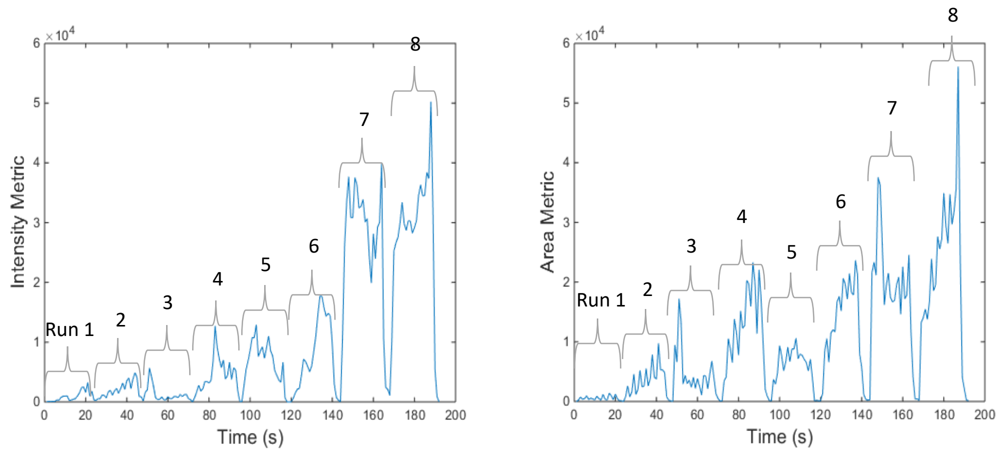
Even though these results and technique were promising for further analysis, there was still the limitation of not being able to implement the area extraction algorithm. Therefore, in order to compare this approach with the still image acquisition methodology in the section 5.2, and to be able to successfully extract spark area, an image combination step was introduced.

Through image combination, most of the information in each image could be considered and packed into a single frame. Also, the erratic variance of the spark and the problematic images with no spark could be overcome. Consequently, after analysing different methods and programming commands available for image combination, an arithmetic summation and averaging of the images was selected for its reliability and processing time efficiency. This method is described by Equation 5.2.

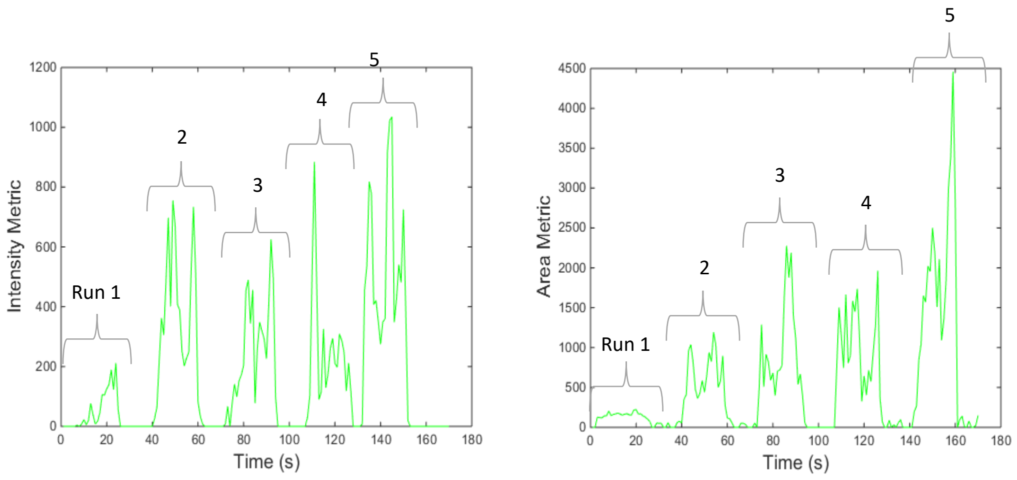
$$Y = \frac{1}{k}\sum_{k=1}^{k}[I(k) - R]$$

Equation 5.2

Where Y is the resulting image, *I* represents each image used in the combination, *R* is the reference image for background elimination (described in the chapter 3) and *k* the total number of images. In Figure 5.7 (a) & (b) this image combination step was implemented, using a sample length of 300 images per combination. This value was selected in accordance to the Moving Average, also delivering a good direct relation with machining time.



(a)



(b)

Figure 5.7: Spark evolution in Test 1: Intensity (left) and Area (right) (a), and Test 2: Intensity (left) and Area (right) (b).

Both, the moving average and the image combination algorithms seemed to successfully emulate the result of a conventional camera with a slow shutter speed. Furthermore, the image combination approach proved that an increased sample length produced richer images with a more consistent spark, easier to isolate and process. Evidently, there is a limit to the length of the sample, which in the present case was set to 300 images, as over-averaging could create a loss of important spark information. On the other hand, this image combination approach made clear that the use of a high-speed feed was unnecessary. The frame by frame processing provided high frequency and noisy results, and the image combination made such a high frame rate redundant. Also, even though the video feed was segmented in frames, which could arguably be like acquiring still images as in the first experiments, the sample rate from the video feed is a clear advantage. Still picture cameras can be very dependent on the internal capacity of the camera hardware and memory speed, which in the first experiments lead to the wide 5 second gaps between images. Video feeds, on the other hand, produce higher sample rates, ranging from around 20 to 60 frames per second in conventional video cameras.

Therefore, it can be concluded that for the present vision-based TCM system, a video feed acquisition system with a sample rate of at least 20 frames would be optimal. On the other hand, the video feed obtained in the second experiments was grayscale, while in the first experiments it was in full colour. The distinctive colour gamma of the spark aids greatly in the spark isolation section of the algorithms. Consequently, another important requirement for the optimal image acquisition system would be to include a full-colour feed. All this proves that a conventional, low-cost video camera device could be used, improving the effectiveness of the general TCM system.

5.4 Summary

The objective of the present chapter was to evaluate and select the best image acquisition system and its optimal parameters. In the first sections, a comparison of different camera shutter speeds was carried out, showing how slow speeds yielded readable and accurate results. However, a highly important limitation was identified, as photographic devices taking still pictures are highly dependent on hardware capabilities, creating in those experiments wide gaps in the acquired data. This limitation was solved by the introduction of a video feed, and the use of a high-speed camera enabled the analysis of different approaches related to the speed of the acquisition parameters.

The next chapter presents a study of the spark-wear relation, where data was used from the two experiments explored in this chapter. Even though the conclusions drawn in this chapter suggested that the first experiments' acquisition system was not optimal, the spark-wear results are important to support the research's hypothesis. Later on, the second set of experiments were analysed, using the optimal results obtained in this chapter, to further support this hypothesis.

6 SPARK EVOLUTION AND TOOL WEAR RELATION

This chapter explores the core of this research, in that it addresses the main hypothesis where the entire study is based: the relationship between the continuous wear that ceramic SiAlON cutting inserts experience during high-speed milling machining, and the evolution of sparks created by this metal cutting procedure. Furthermore, this chapter is greatly based on a conference publication by this work's author (Dominguez Caballero et al. 2016b), where all the following statements were presented.

The data obtained by the author's previous work and described in chapter 3 as the first set of experiments were revisited for this chapter. Algorithms were updated and optimized, and new algorithms were created to generate a more conclusive evidence of the research's main hypothesis.

The second set of experiments was also used to evaluate the hypothesis in this chapter, where evidence of the spark-tool wear relationship was also found.

6.1 Methodology

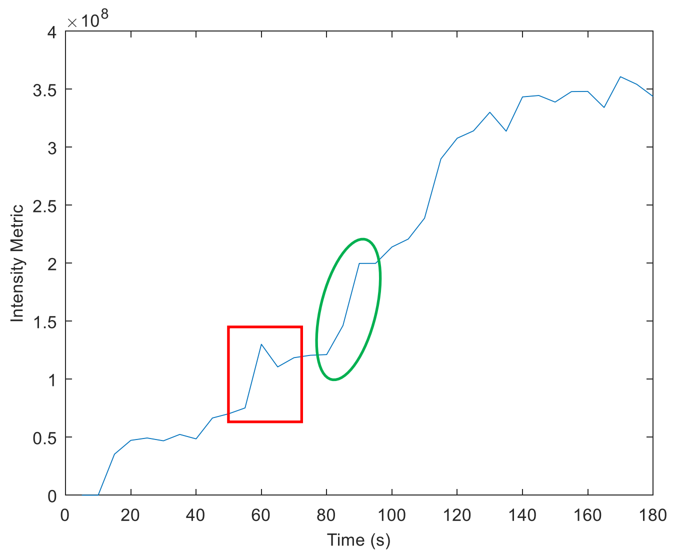
The first set of experiments was discussed in detail in the chapter 5, and therefore will not be once again fully described. Yet, it would be relevant to remember that an SLR camera was used in four test that included high, medium and slow shutter speeds. However, in the present chapter, only the fourth test will be assessed. This is because the fourth test included machine stops to measure tool wear, as well as slow shutter speed, given the conclusions that were drawn in the previous chapter.

In the second set of a high-speed video camera was used to record the cutting sparks. It is important to remember from the methodology chapter that in test 1, the ceramic insert was extracted from the tool then weighed and photographed between runs. In Test 2 however, the insert was also photographed, but this was conducted with the insert still mounted in the tool and consequently, no mass measures were extracted. For this reason, only the results from test 1 of this second set of experiments will be explored in the present chapter. Furthermore, the image combination algorithm will be used as these tests included a video feed.

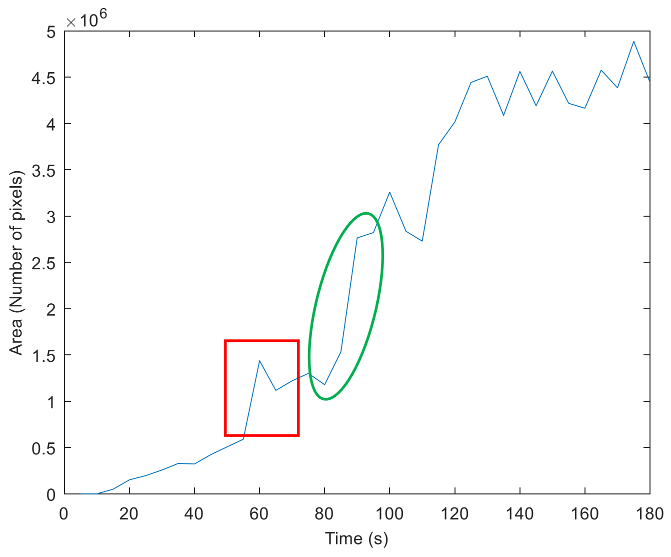
6.2 Spark – Wear in First Experiments

6.2.1 Spark Feature Results

The images obtained through the fourth test in the first set of experiments were downloaded into the MATLAB software for processing. After this, the two algorithms described in chapter 3 for extraction of intensity and area were implemented. Also, as mentioned there, the cutting process was divided in 8 runs where the process was stopped to measure tool wear. This presented a problem when analysing data from the spark algorithms, as the stops created low or zero values of intensity and area. For this reason, an extra processing step was introduced to remove those sections of data where these stops occurred. Figure 6.1 shows the resulting spark evolution in intensity (a) and area (b) (see Appendix V for the raw data with the machining stops). Both graphs have, on the horizontal axis, time in seconds, regarding machining time of the milling process. The vertical axes, on the other hand, are for intensity metric in the first and area in number of pixels in the second. As may be expected, both figures display some similar behaviour. There is a sudden increase in spark intensity and spark area at 60 seconds, highlighted by a red square. Also, there is a similar high gradient between 80 and 100 seconds, encircled in green. Both figures also display a similar gradually-increasing trend of the descriptors throughout the machining time. This shows that these descriptors are correlated and are representative of the spark development.



(a)



(b)

Figure 6.1: Spark evolution: intensity (a) and area (b).

6.2.2 Insert Wear Results

In this set of experiments, only microscope images of the rake face of the inserts were recorded. This was one of the main changes that were introduced during the following experiments after the realisation of how it limited the quantity of wear data. However, since these first experiments were performed in collaboration with the AMRC (Advanced Manufacturing Research Centre), the author had little influence in how the tests were conducted.

For the rake face images, the procedure described in the section 4.1.1 was used, and crater wear area was extracted from each insert's image. The results are shown in Figure 6.2, where the trend starts from time zero, where the insert has no wear marks, up to its maximum value of 12.11 mm^2. It is important to mention that this value of maximum crater wear area is in accordance with the results obtained in the wear assessment chapter.

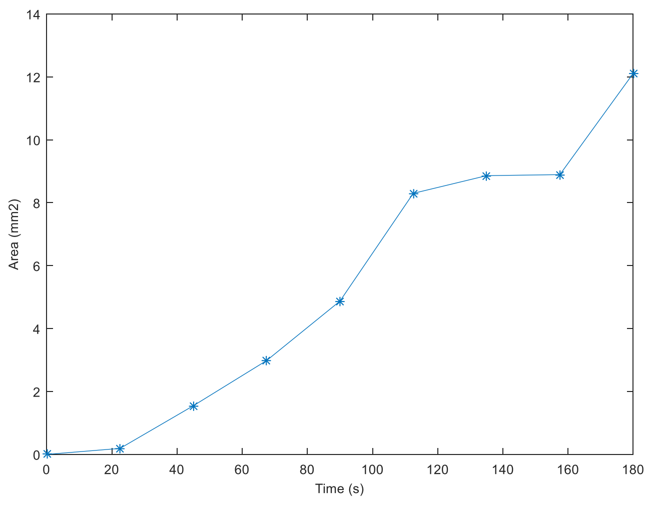


Figure 6.2: Insert's crater wear area.

6.2.3 Spark – Tool Wear Relation and Discussion

Figure 6.3 shows both spark descriptors along with the wear measurements of crater wear area. The vertical axes are applicable for each value graphed in the figure, where the left side vertical axis in blue corresponds to the spark intensity values, also in blue and with a dashed line. On the right, there are two vertical axes, the one in black and nearest to the figure corresponds to the spark area, with its trend graphed also in black, with a dashed line and markers for each recorded value. Finally, the vertical axis in red and in the far-right corresponds to the insert's crater wear area, found in a solid red line inside the figure.

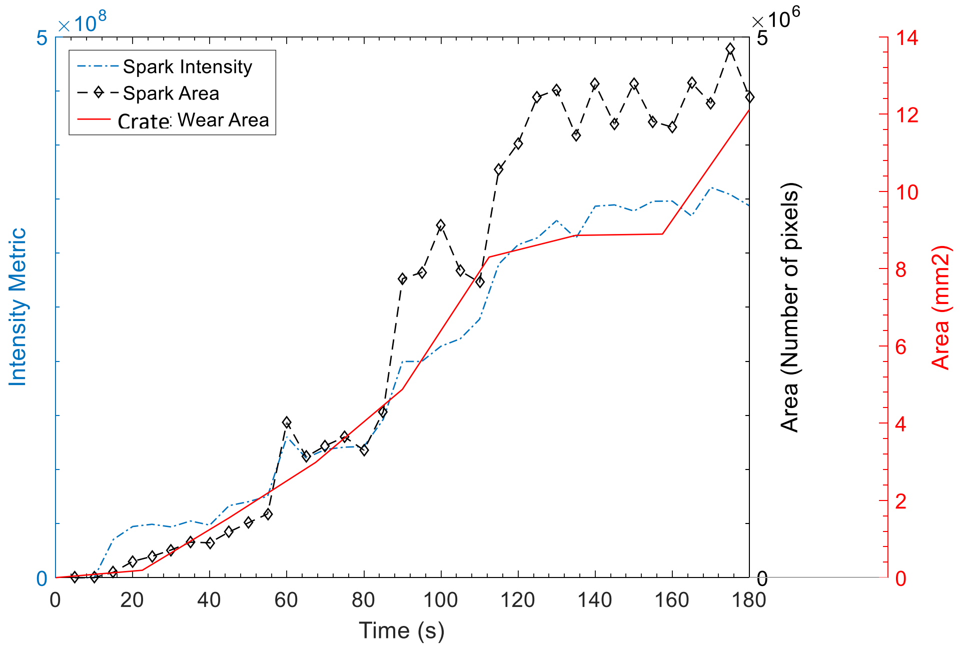


Figure 6.3: First experiments - Test 4: spark area, spark intensity and insert's crater wear area.

In the figure, it is possible to see that all three measures have a general qualitative correlation, as the steepness of them all is very similar, particularly between zero and 120 seconds of machining time. However, it was impossible to calculate a quantitative correlation of all these trends, as there is a dimensional mismatch between the number of data points in each one. While both spark descriptors have the same amount of values, the crater wear area measurement only includes 8 points. It could be argued that some curve fitting could be done between these data points to calculate correlation values, but given the nature of the crater wear area (described in section 4.2.2), it was decided not to do it at this point of the research.

On the other hand, the optimised algorithms for the area and intensity extraction appeared to be working correctly in isolating and extracting spark data. The trend lines of both descriptors seemed to run smoothly and with very little presence of noise. The expected general and gradual increase of both features with machining time occur, supporting that these descriptors are indeed most relevant for spark behaviour analysis. Furthermore, the tool crater wear extraction algorithm and results also seem to be creating a clear and smooth path throughout the machining time.

Another limitation that was encountered was the image acquisition system. As it was described in chapter 5, the use of an SLR camera taking still pictures every five seconds created very wide gaps of data in the spark evolution behaviour. While conducting the experiments, interesting variations of spark evolution could be spotted with the naked eye, which were lost in the normalisation that these gaps create. Furthermore, there was a lack of other tool wear mechanism measurements; and whilst the crater wear area was successfully extracted and related to the other descriptors, more wear data would have been certainly beneficial.

Regardless of these multiple limitations, the intrinsic similarity of these three resulting trends successfully supports the basic hypothesis of this research, enabling the further exploration of the desired tool monitoring system using this phenomenon. For this reason, and given the limitation already mentioned of the dimensional difference of the variables, a predictive scheme using neural networks was chosen for the next step in the research. As it was described in the literature review chapter, these machine learning tools can absorb dimensional differences in variables given the robustness they can attain if the correct number of layers and weights are chosen.

6.3 Spark – Wear in Second Experiments

The second set of experiments attempted to overcome some of the limitations described previously by implementing the continuous high-speed video feed and introducing in the first test the weighting of the cutting tools. Therefore, this section has the objective of supporting the conclusions drawn before by giving additional evidence of the spark-wear relationship found in these new experiments. However, as mentioned in section **Error! Reference source not found.**, only the first test will be assessed since it included both weight measurements and microscope images of the cutting inserts. Additionally, only the spark area trend will be used to compare spark and tool wear to improve the data readability at the end of this section.

6.3.1 Insert Wear Results

All the tool wear mechanisms described in the chapter 4 were also found in the cutting tool used in this test. The presence of adhesion and flaking of the inserts rake face was evidenced by the mass loss shown in Figure 6.4, where each marker on the mass loss trend corresponds to a machining run. It can be appreciated how there are small mass loss decrements from run 1 to 2, from run 3 to 4 and from run 5 to 6, showing the presence of adhesion. Additionally, there are clear increases of mass loss from run 2 to 3, from run 4 to 5 and from run 6 to 8, showing the flaking mechanism.

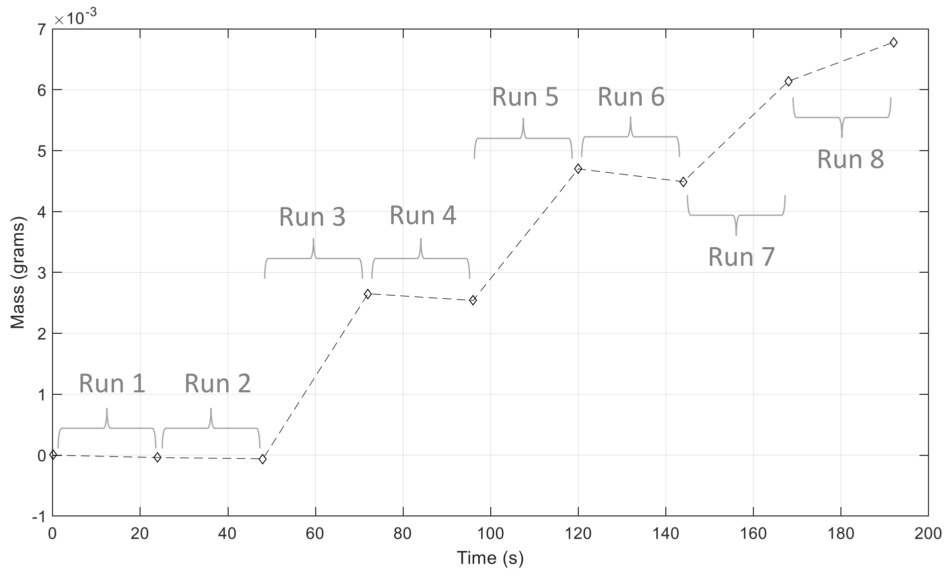


Figure 6.4: Insert's mass loss.

Figure 6.5 shows the resulting crater wear areas obtained by using again the image processing procedure described in section 4.1.1. Comparable to mass loss, the presence of flaking was found in the clear increases of crater wear from run 2 to 3 and from run 4 to 5.

However, it was found that from run 5 to 6 there was a discrepancy in these wear descriptors, as mass loss decreased while crater wear area increased. This was found to be caused by the adhesion mechanism, which occluded the insert's material loss when weighed. This variance shows how the crater wear area assessment can overcome this issue, which could otherwise be misleading.

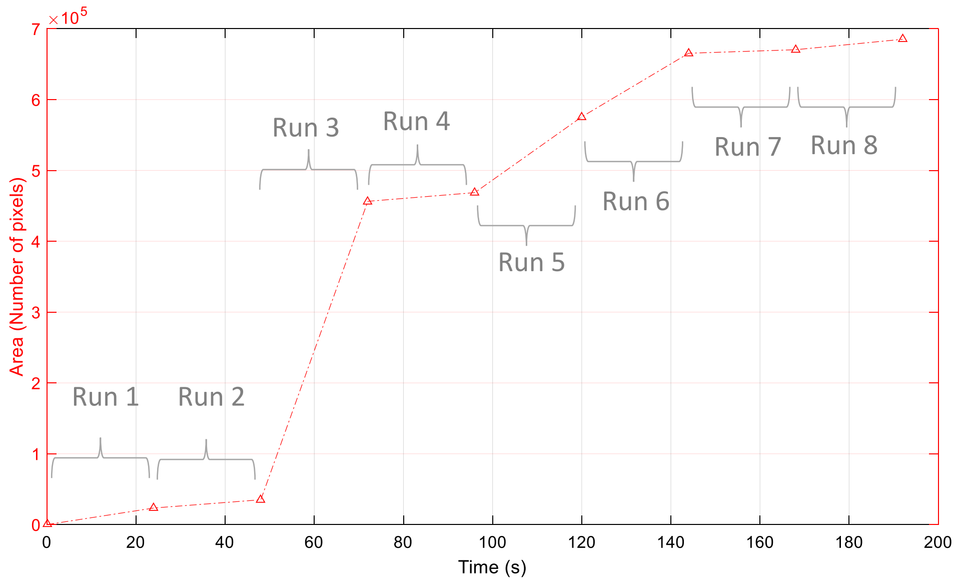


Figure 6.5: Insert's crater wear area.

6.3.2 Spark – Tool Wear Relation and Discussion

At the results of the first set of experiments in section 6.2.3, it was easier to observe a qualitative correlation between spark and wear. The wide gap between images enabled a very gradual behaviour of spark evolution, with a continuous increase. In this second set of experiments, on the other hand, the rise in sample rate provided a more detailed spark evolution, where a similar continuous trend was not easily identifiable. Figure 6.6 shows the spark area evolution in solid blue line (left-hand axis), along with a mass loss in dashed black with diamond markers (first right-hand axis), and with crater wear area in dashed red with triangle markers (second right-hand axis). Although the spark evolution had a generally increasing trend throughout time, it can be seen in the figure that there were plenty of discontinuities in spark area evolution. However, on closer inspection, it was found that these spark "fluctuations" could be attributed to important events in the cutting insert's wear behaviour.

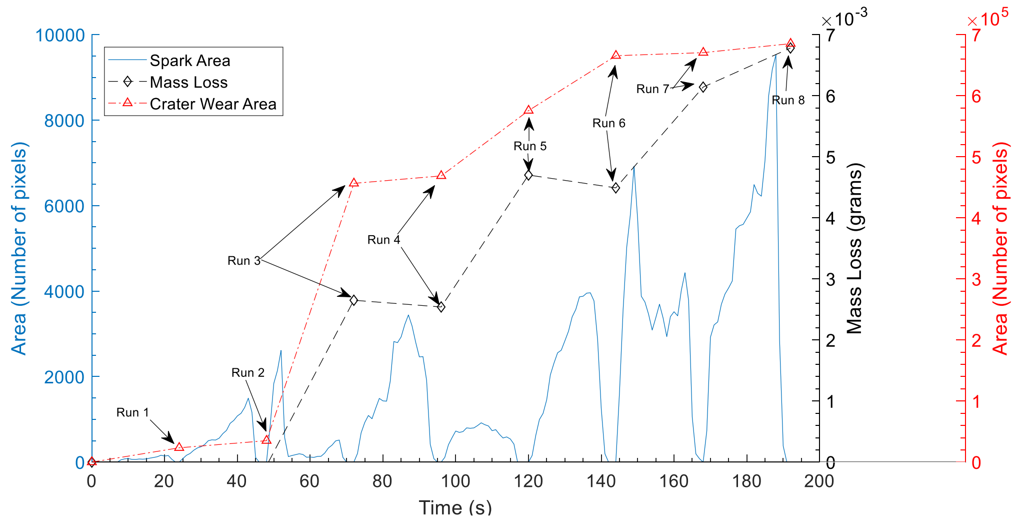


Figure 6.6: Second experiments – Test 2: spark area, insert's mass loss and insert's crater wear area.

The sudden increase of mass loss and crater wear area from run 2 to run 3 coincide with important drops of spark area. This is also the case from run 4 to 5 and run 6 to 7, where spark evolution drops are met with mass loss and crater wear area leaps. Analysing these results, and performing a detailed examination of the video feed, it was apparent that every time the cutting tool suffered a flaking or chipping of material, the spark seemed to rapidly decrease in size. The proposed explanation for such behaviour was that every time the cutting tool lost an important amount and area of material, a new sharp edge was revealed. This new sharp edge would create a temporary new low worn surface of the insert, reducing the size and intensity of sparks, but creating poorer and poorer surface finish. Additionally, the loss of cutting tool material would also impact in the accuracy of the cutting depth and feed per tooth.

This phenomenon of sudden cutting tool material loss appears to be of great importance for the current research. These events seemed to be crucial to accurately monitor the actual tool condition and possible tool failure. It was therefore proposed that these "flaking events" were the failure mechanisms that could lead to tool breakage. As was described in the chapter 4, flank and notch wear were found to be important mechanisms to continuously map the tool's condition, even though they did not seem to be the main causes of tool failure. Nevertheless, as it was also described in that chapter, these two mechanisms are significant precursors of the microcracks that appear to weaken the cutting tool material and lead to the mentioned tool flaking and chipping.

6.4 Summary

It is almost impossible to accurately and constantly measure the tool wear of cutting tools whilst machining, otherwise this research and many other mentioned in the literature review would be unnecessary. Therefore, the different ways in which indirect methods of TCM are compared with actual wear can always appear to be quite empirical. However, a well-supported method of comparing and aligning these two can be successful when enough evidence of correlating behaviour and event recognition can be discovered.

The present chapter had the already mentioned objective of exploring and studying the main research hypothesis. The two experiments chosen accurately proved that this hypothesis has a very strong foundation and that further work was possible. The first experiments accurately showed that there was an intrinsic qualitative correlation between tool wear (crater wear area) and spark evolution (area and intensity). The second experiments, on the other hand, expanded this correlation by showing the presence of spark events that could be directly related to tool wear.

Even though these correlations could not be fully quantified due to the dimensional discrepancy of the data in those tests, other methods could overcome this by using a more robust method of fitting spark behaviour with wear data. This is the case of Neural Networks, which are effective models of pattern recognition through statistical analysis. Fed by a compound of experimental data as training, neural networks can accurately obtain a compact model, with fast data processing (Bishop 2006). Therefore, the next chapter will present the design and application of a neural network for the prediction of tool wear. This neural network was entirely based on the conclusions drawn by all of the previous chapters. For instance, based on the conclusions exposed in the chapter 5, the thrid set of experiments incuded the use of a video feed to implement the image combination algorithm, emulationg the mentioned "slow camera settings". On the other hand, the selection of output variables of the neural network came from the assessments and conclusions drawn in the chapter 4. And finally, the general use of variables and the interpretation of results were mainly based on the conclusion and discussion presented in this chapter.

7 PREDICTIVE TOOL CONDITION MONITORING VIA NEURAL NETWORKS

All the previous chapters included different objectives that together, had the goal of supporting the decisions included in this final chapter. Having shown in chapter 6 that there is an apparent empirical or qualitative correlation between the spark evolution and the measured cutting tool wear, it was decided to attempt to implement a method of tool life prediction. As was described in the literature review in section 2.4, Artificial Neural Networks (ANN) are mathematical models which are capable of representing input-output relationships via a process of pattern recognition and supervised learning. Therefore, the current chapter describes the implementation of this model to the data extracted from the final experiments, with the objective of building a neural network capable of predicting tool wear.

7.1 Methodology

In section 3.3.3, the third set of experiments was described, where a design of experiments (DOE) was used to plan different tests, varying different relevant machining parameters. It was mentioned then that the conclusions found during the analysis of the first and second set of experiments were essential to the planning and execution of these final tests. It was found through the assessment described in chapter 5 that the best image acquisition feed for this research would be a video feed, and that the image combination algorithm as a method of simulating slow acquisition settings was fit for purpose. In chapters 4, the third set of experiments was also used, given the richness of data obtained from those tests. However, these tests were not revisited or described, as it was deemed important to do this in the present chapter, where all the data extracted from these tests would be used. Nevertheless, in that chapter, as well as in the literature review section 2.1.2 and chapter 6, the importance of certain tool wear values was stated, and this was also an important factor for the design of that third set of experiments. Therefore, as it was described then, the DOE included different machining parameters organised into orthogonal designs to reduce the number of tests whilst achieving a good mix of values. The collected wear datasets have already been presented and described in chapter 4, however, the area and intensity datasets have not been discussed. These two features had to be extracted from the many video recordings of each machining run inside each test.

As was described in section 3.2, the general intensity and area extraction algorithms followed a similar structure and order throughout all the experiments, however, these had to be adapted for each set of experiments. In the case of the present third set of experiments, there were two major corrections to the general algorithm, where the subtraction of a reference image from all frames was eliminated, and an additional step to remove data from time periods when the tool was not engaging the workpiece was introduced.

Firstly, the elimination of the use of a reference image to reduce background information presented a problem during these experiments. Given that the workshop where the tests were conducted was relocated, the new location presented a problem that the previous sets of experiments did not experience. As was mentioned a few times in chapter 3, the external illumination conditions in the first and second sets of experiments were kept as constant as possible. Nevertheless, the new location of the milling machine included direct incidence of natural light, and hence, different illumination conditions throughout the day and the seasons. Even though minimisation of this effect was attempted by reducing the ISO parameter (also described in chapter 3), as well as by partially blocking windows, this variation was observed in all the videos. The way this variation caused a problem with the existing algorithm was that, since the reference image was the first image of the data compound, this picture would generally have a very different illumination that the rest. Consequently, as the reference image subtraction was an arithmetic subtraction of intensity levels of the images, in some cases this would not eliminate any of the background, producing unreliable spark intensity and area data results. Therefore, this step was replaced by a different approach in the general algorithm. Instead of subtracting a reference image from all the frames, each frame was subtracted from itself by using an image morphological opening technique. Each frame was partitioned in each of its RGB channels, as shown in Figure 7.1, and given that the spark is mostly present in the Red channel, this technique was applied only to the other two channels.

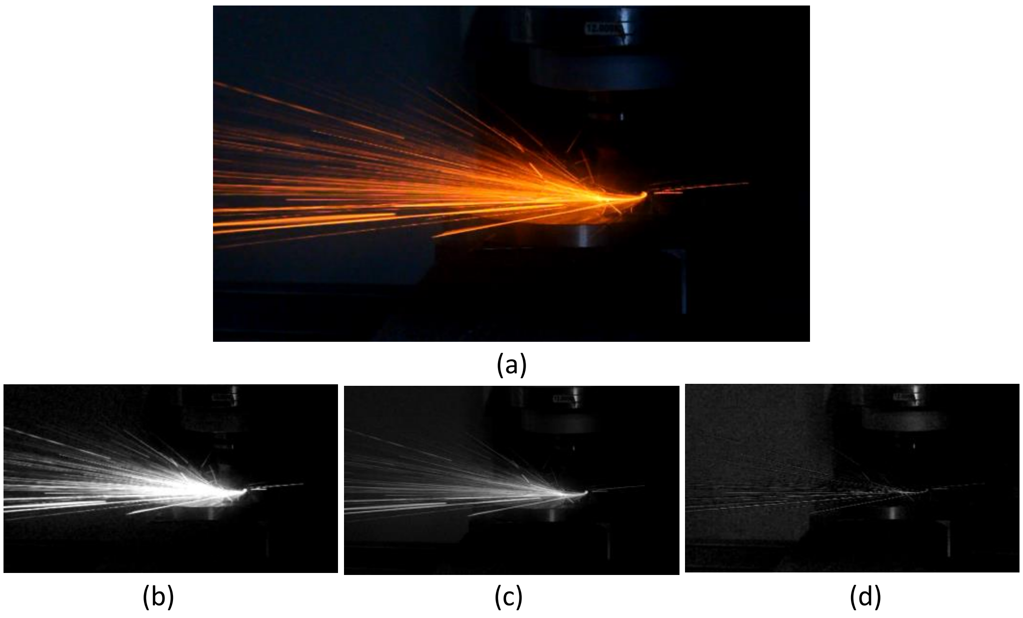


Figure 7.1: Partitioning of original image (a) into Red (b), Green (c) and Blue (d) channels.

Through the morphological opening technique, the image's pixels experience an erosion followed by a dilatation, based on a determined structuring element. The structuring element represented a neighbourhood of pixels, limited by a certain distance and a certain shape, where this erosion and dilatation would be implemented. The erosion consists of mapping structuring elements so that the centres of these elements are distributed throughout the image and small image elements inside are eroded. The dilatation mechanism does the opposite, using again the structuring element, but incrementing the area of picture elements. In the present approach, a "disk" shaped element was chosen, as it is an element commonly used for these tasks, for it improves processing time whilst its shape has distinct advantages for background removal. With these elements, the radius determines its size and another value is used to approximate the disk shape. After this technique was applied to the Green and Blue channels, these were concatenated to reform a three-channel image, but with a structure BGB, as shown in Figure 7.2. This last image is the one that is subtracted to the original image and the resulting effect can be observed in Figure 7.3, where most of the background has been removed and the spark has been isolated.

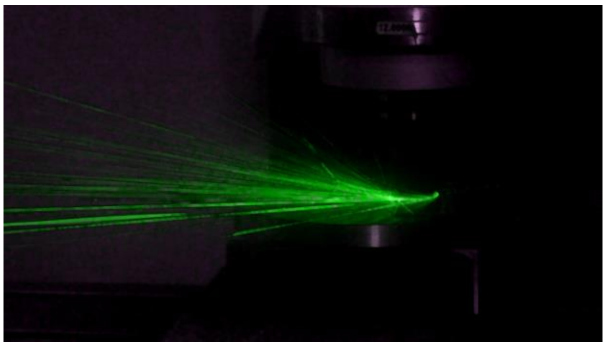


Figure 7.2: Concatenated BGB image from the morphological opening of the green and blue channel.

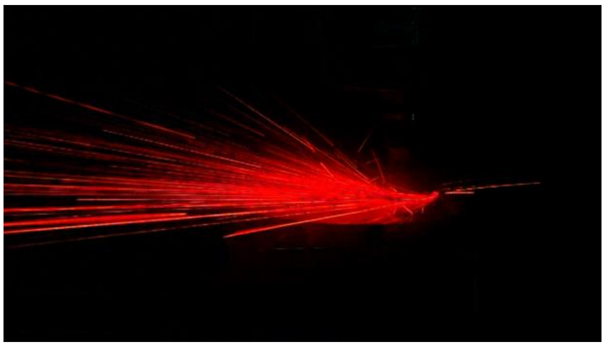


Figure 7.3: Resulting image from subtraction of concatenated GBG image.

The second adjustment to the general algorithm methodology was the introduction of a step to remove the spark area and intensity values obtained from frames where the tool did not engage the workpiece, as shown in Figure 7.4 where these values are encircled. During testing, the video feed was initiated as the machining run was started, and since the cutting tool would travel some distance before engaging the workpiece, those frames were also recorded. Likewise, at the end of the run, the cutting tool would exit the workpiece and frames would include small sparks from the disengagement and no spark from the remaining tool path. This was the same situation as in the second set of experiments, where all throughout section 5.3 the resulting graphs would have a fluctuating outline that showed each machining run. However, in the present set of experiments and for this chapter's objectives, it was important to eliminate these apparent "low values" of spark intensity and area, as they could be problematic for the neural network. Therefore, this step would evaluate the area and intensity results extracted from each video (each run), calculating the standard deviation of their values. Then, the first value found to be above a certain percentage of the standard deviation would be located and regarded as the first value of actual tool – workpiece engagement. The percentage value was introduced to give this discrimination step more flexibility, and it was set to 80% after repeated trials as the spark behaviour varied significantly from test to test. After this, a certain number of the following values was recorded, ending this window with the last value of actual tool – workpiece engagement. Since this task did not need to be automated, the selection of this window of values and the evaluation of the last value was done manually, mainly to avoid loss of relevant data. Also, the window of values was the same for all the runs in the same test, as they all had the same toolpath, and hence, the same machining time. The full algorithm for this set of experiments can be found in Appendix VIII.

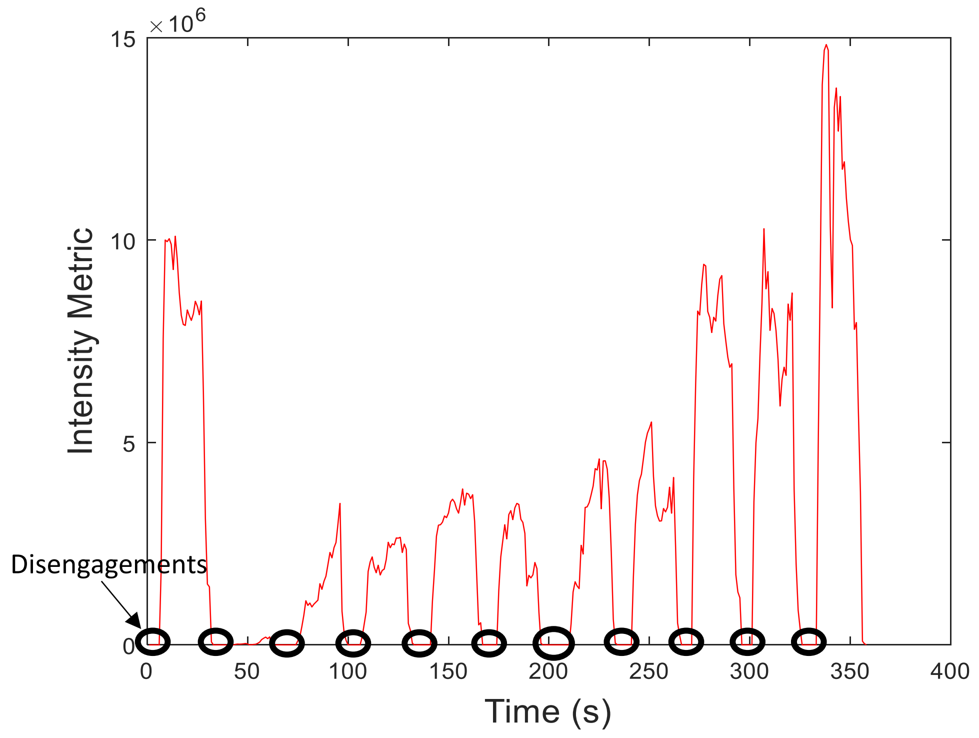


Figure 7.4: Spark intensity in Test 1, third set of experiments, with tool disengagements encircled.

In section 4.2.3, the tool wear measurements of all the tests were presented. However, it is important to mention that during data processing it was discovered that the video recordings of test number 15 were missing; and whilst its tool wear measurements were recorded successfully, this test had to be omitted during this chapter. Retesting was considered but, given that the workpiece material and cutting tools had been completely used, preparing new material and purchasing more tools for one test was not deemed a good use of resources.

7.2 Two-Layer Neural Networks and Netlab Overview

As was mentioned in the literature review, in section 2.4, the Multi-Layer Perceptron is a widely used architecture of a Neural Network (NN) for its adaptability and its processing efficiency. This architecture generally consists of two layers of weights interconnected between the nodes, as shown in Figure 7.5. Furthermore, this architecture is generally capable of "universal approximation," as they can approximate any function with a certain level of accuracy, assuming there are enough hidden nodes (Nabney 2002). In Figure 7.5, from left to right, the first layer corresponds to the *input layer* of variables x_i where $i = 1, \dots, D$. Then the second layer corresponds to the *hidden layer*, with its hidden units or hidden nodes z_j with values $j = 1, \dots, M$. The variables x_0 and z_0 correspond to the bias parameters associated with these two layers. Finally, the last layer corresponds to the *outputs layer*, denoted with the variable y_k, where $k = 1, \dots, K$.

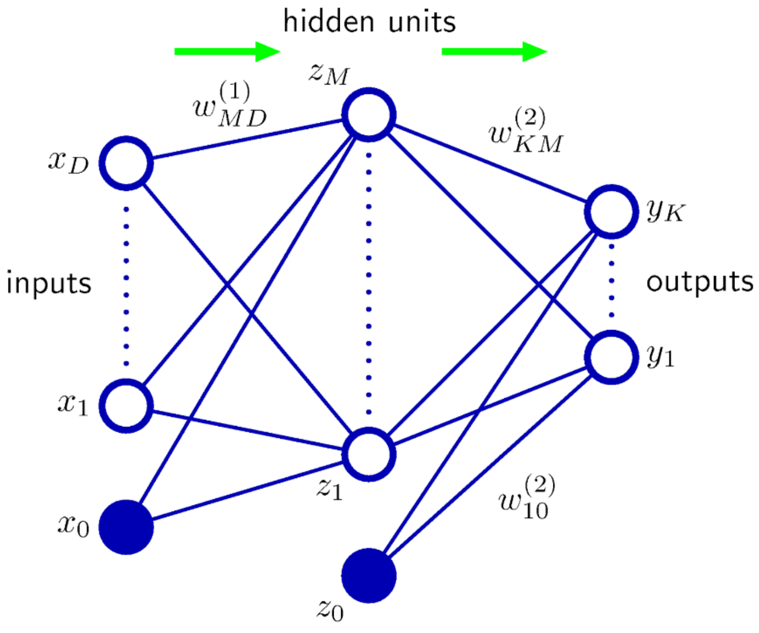


Figure 7.5: Diagram of a two-Layer feed-forward architecture of NN (Bishop 2006)**.**

In any neural network, the weights w_j that are interconnected between layers are the ones that are adapted throughout the training phase, as they act during the *feed-forward* propagation. This propagation is carried out by a set of intermediate activation variables that, from the first to the second layer, are given by Equation 7.1. Every $a_j^{(1)}$ variable is associated with each hidden unit and the variable $b_j^{(1)}$ correspond to the bias parameters associated with the hidden units.

$$a_j^{(1)} \sum_{i=1}^{D} w_{ji}^{(1)} x_i + b_j^{(1)} \qquad\qquad j = 1, \dots, M$$

Equation 7.1

After this, the activation variables are transformed using a non-linear activation function at the hidden layer, and their outputs are given by Equation 7.2.

$$z_j = \tanh\left(a_j^{(1)}\right)$$

Equation 7.2

Finally, the z_j is then transformed into a second set of activation values by the weights and biases in the second layer, given by Equation 7.3.

$$a_k^{(2)} \sum_{j=1}^{M} w_{kj}^{(2)} z_j + b_k^{(2)} \qquad\qquad k = 1, \dots, \mathrm{K}$$

Equation 7.3

For regression problems, which is the case of this study, the output layer variables take the form of the linear function $y_k = a_k^{(2)}$. In the case of classification problems, the activation output units are transformed using other different functions that will not be discussed in this work.

As mentioned previously, neural networks are capable of approximating continuous functions by supervised learning through a set of training data points. The way a NN's training phase enables the adaptation of the weights between layers is a technique named *backpropagation*. This is a complex technique that is repeated during training by a defined number of iterations, until the NN achieves in approximating, to a certain accuracy, the input-output relationship. Initially, the NN model assigns random values to the weights. The obtained output values are then compared to the already known *target* values t_k in the manner given by Equation 7.4, for each pattern n in the data set. After this, error values δ are calculated and through error backpropagation, the weights are updated to a new value. This process is repeated a certain number of times as the error gets reduced; arriving at an input-output function approximation with a certain level of error.

$$\delta_k^{(2)n} = y_k^n - t_k^n$$

Equation 7.4

To find the best neural network for a certain set of input-output data, it is important to select the correct number of hidden nodes. If the number of hidden nodes is too high the network can become overcomplicated, impacting in processing efficiency, and most importantly, in possible "overfitting" of the data. In the other hand, if the number is too low, the network may be underfitting data, delivering a low level of accuracy. Overfitting is a common problem in the implementation of neural networks. When this issue occurs, the networks may appear to deliver low values of error for the points used during the supervised learning, but when new data is presented, the error becomes very high.

The common practice to build and select a fully functional NN is by a cross-validation method, where the input data is divided into three data sets, namely for Training, Validation and Testing. The way data points are divided can be arbitrary, depending on the type of input data and the purpose of the NN; similarly, the number of data points for each section of the cross-validation can also be selected arbitrarily. However, for the latter, the number of training points can generally be around the 70% as it is desirable to use most of the examples for the supervised learning but leaving some instances for validation and testing. Consequently, the number of validation and testing points can be around 15% each. Later in this chapter, these quantities will be selected as per the analysis of the type of data used.

The software used to create NN in this research was a toolbox for MATLAB software named Netlab[2] by Nabney (2002). This toolbox includes different commands and operations that aid in building and cross-validating a NN. As an overview, to create a NN in this toolbox, the initial step would be to set-up the network parameters, declaring the number of inputs, outputs, hidden nodes and *hyperparameters*. Then, through a separate command, the network's structure is initialised using all the parameters previously declared. After this, some optimising parameters are introduced to control the NN, where the most relevant parameter relates to the desired number of training cycles. Finally, the algorithm uses another command to optimise or train the neural network using some input values and their respective known target values. This trained network would be the one used in the validation step to select the best network and avoid overtraining, and in the testing step to report the NN final error value.

Even though this overview mentions the general procedure of the Netlab toolbox, there are some commands, factors and operations that are worth analysing before the NN can be implemented on the research's data. Firstly, there are different optimisation algorithms that can be applied to the main operation of "network optimisation," which is the Netlab command that trains the network. In the present research, the optimisation algorithm selected is the "Scaled Conjugate Gradient," given that this algorithm is widely used in two-layer neural networks in MATLAB. According to Nabney (2002), it can provide conjugate searched directions, without performing line search or calculating the Hessian matrix, and it can outperform other conjugate gradients and quasi-Newton algorithms.

[2] Netlab Neural Network Toolbox by Nabney and Bishop, Aston University, Birmingham. URL: http://www.aston.ac.uk/eas/research/groups/ncrg/resources/netlab/

Furthermore, before building and optimising the NN the hyperparameters, as well as the number of training cycles or iterations, can be declared. For the NN built in this chapter, only the *weight-decay* hyperparameter $\propto$ was declared. It is a regularisation parameter used in Netlab's neural networks to penalise large values of weights, and hence avoid overfitting. On the other hand, the number of training cycles declared is the number of times that the optimisation procedure will be carried out. For both parameters, it is advised to use cross-validation to select their values; however, the amount of time required, compared to the possible benefit of doing this long study was not deemed necessary by the author. Therefore, a fixed $\propto$ value and a fixed number of iterations were selected and implemented in all neural networks, and their values were selected by recommendation of the Netlab literature.

Another set of commands and values worth analysing are the error values that the toolbox algorithms calculate, as these are relevant for the presentation of results later in this chapter. During the training cycle of the NN, each iteration automatically calculates and records a *Bayesian error* value. This type of error value is widely used in Netlab, where is also known as "overall error," as it uses the regularisation parameter $\propto$. The value of this Bayesian error is given by Equation 7.5. The variable E_D is a "data error" calculated in Netlab by Equation 7.6, where N is the number of data points; and E_W is a "weight error" term in the form of Equation 7.7 inside the toolbox, where W is the total number of weights.

$$E = E_D + \propto E_W$$

Equation 7.5

$$E_D = \frac{1}{2}\sum_{n=1}^{N}\{y(x^n; w) - t^n\}^2$$

Equation 7.6

$$\propto E_W = \frac{\propto}{2}\sum_{i=1}^{W} w_i$$

Equation 7.7

Given that the recording of the Bayesian error is done automatically throughout training, this type of error will be used to present training results. On the other hand, to present all other results of the cross-validation process, the mean-squared error (MSE) will be used instead, which can be calculated using the data error by implementing Equation 7.8. The MSE is used in most literature regarding neural networks' performance, and therefore, this was deemed the most appropriate for this work.

$$MSE = \frac{2E_D}{3N}$$

Equation 7.8

In the following sections, the different decisions regarding data manipulation and NN algorithm implementation will be presented and discussed. Therefore, it would be relevant to relate back to this section to understand the different concepts used for the neural networks and for Netlab.

7.3 Neural Networks and Results

In the present section, the different neural networks that were created and the cross-validations that were carried out for the third experiments' data will be presented. In an ideal set-up, a single NN would be created using all the data obtained in those experiments. This would mean using some tests for training, some for validation and some for testing. However, to successfully understand and analyse the neural network's behaviour, it was decided to divide this section into three steps. Firstly, to see whether it was possible to relate the spark features as network's inputs to the wear parameters, this step consisted in carrying out a cross-validation for each test individually. This permitted an isolated analysis of tests' results, making possible to understand the behaviour of the networks and their respective levels of accuracy. After this, the machining parameters were included in a cross-validation with all the tests' data to observe their impact on the neural network. However, in this step, the tests' data points were randomly selected and assigned to the training, validation and testing data sets. This attempt permitted the creation of a single NN for all the tests and with data points from all the tests inside the training, validation and testing steps, enabling a more direct interpolation of the targeted outputs. Finally, a cross-validation was carried out using again the entire tests' data, such as in the previous step. However, all the points of each entire test were taken together into the training, validation and testing data sets; such as it was described above as the ideal set-up of the NN. The objective of this approach was to evaluate whether a single trained NN could correctly predict the wear outputs using data from unseen tests. Consequently, the sorting of these tests was done using the DOE orthogonal designs, which had certain properties that were useful for this final approach. Therefore, in a summary, the first individual step helped to understand the challenges that the NN could face by isolating their implementation, whilst the other two steps provided a good idea of how these networks would work in a more demanding scenario. Nevertheless, some challenges were found during each cross-validation, and some changes were required and implemented.

Some parameters were kept constant in all the three steps; these were the optimisation algorithm, the hyperparameter, the number of training cycles and the use of error values. Furthermore, regarding the number of data points for each cross-validation section, it is important to mention that neural networks tend to have a better performance when a high number of examples and data points are used during training. However, given that a DOE was conducted to minimise the number of tests due to resource and time limitations, the total number of examples and data points was not very high. Therefore, it was decided to assign 80% of the data points for training, allowing a high number for the supervised learning phase of the optimisation. Hence, 10% of the data points were assigned for validation and 10% for testing.

The input and output variables chosen for the networks can be found in Figure 7.6.

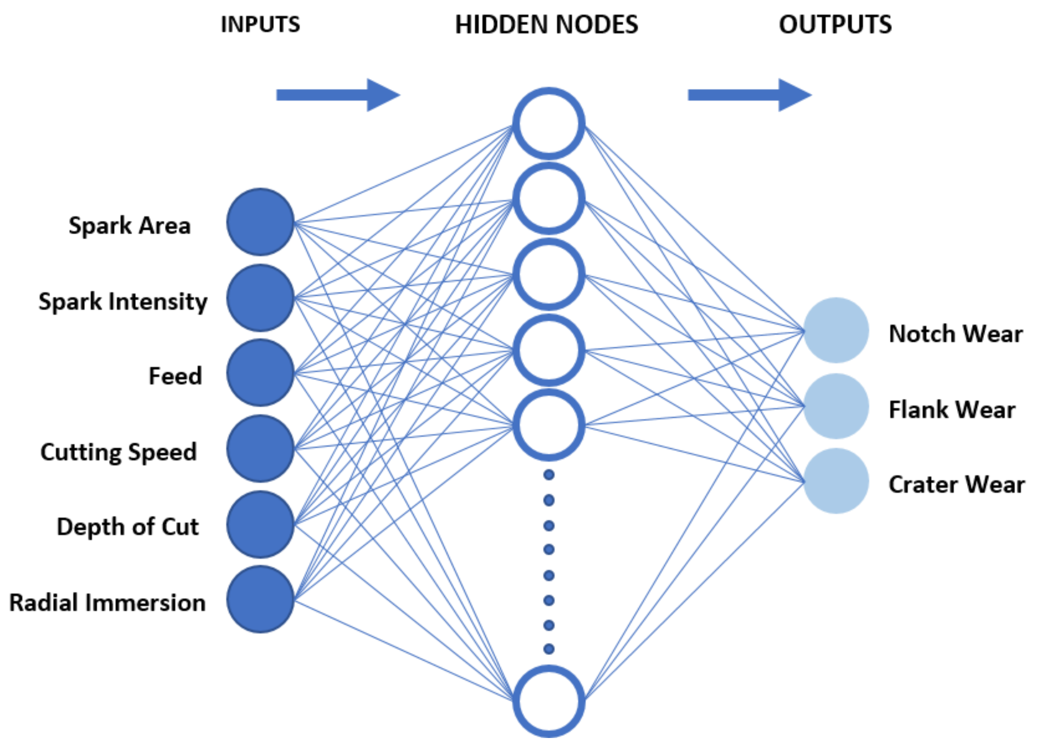


Figure 7.6: Two-layer neural network for machining spark – wear prediction.

As can be appreciated, the neural networks inputs include the spark features and the machining parameters. In a real-time scenario, these parameters are the only factors that the NN could know during the machining process, as the spark features could be processed in real-time and fed into the NN. On the other hand, the outputs are the wear measurements discussed and presented in section 4.2, where their importance was described, whilst their relationship with spark features was already explored in chapter 6. However, the wear measurement of mass loss was not included here, given that it was not a continuously increasing parameter, where adhesion acted to give potentially problematic gains of mass. This could affect and confuse the NN, and it was decided that the crater wear area was also capable of providing the information that the mass loss parameter could provide.

Furthermore, it is common practice to normalise the data points fed into a NN to avoid unbalanced networks and large weights. Therefore, all the inputs and outputs were normalised using a MATLAB resource that centres the data to have a mean at 0 and scale the data to have a standard deviation of 1. This can be appreciated in Equation 7.9, where μ is the mean and σ is the standard deviation. Thus, all the results in this chapter will be presented in normalised values.

$$z = \frac{x - \mu}{\sigma}$$

Equation 7.9

Finally, the algorithms built for these networks followed the structure shown in Figure 7.7.

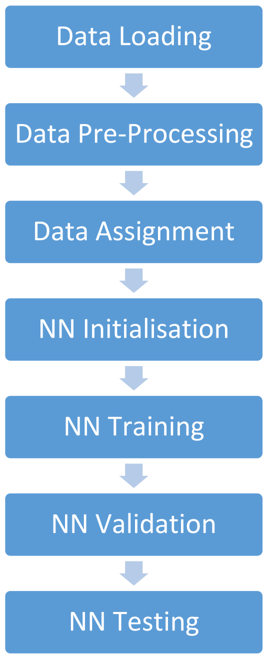


Figure 7.7: Neural network algorithm for machining spark – wear prediction.

Where each step has the following description:

1. **Data Loading:** In this step, all the data that was collected in the third set of experiments is loaded into the algorithm (Spark Area and Intensity as inputs, as well as Flank, Notch and Crater Wear as target values).
2. **Data Pre-Processing:** The pre-processing task included the regularisation of variables, followed by a curve fitting of the wear parameters. Given that wear measurements were carried out between runs, only 11 data points were recorded for each test. Therefore, the curve fitting step created the additional points required to match the spark features dimensions. In all algorithms, a linear interpolation was used for curve fitting, as other methods such as cubic interpolation were tested, but

their results were not adequate. The final processing of this step included the creation of the machining variables, assigning values of -1, 0 and 1 to each level of the variable (see section 3.3.3.1).

3. **Data Assignment:** In this step, the data points of the test or tests were assigned to a different cross-validation section (Training, Validation, Testing). This step was different for each one of the following approaches, and its implementation will be described in each section.
4. **NN Initialisation:** This step includes all the sub-steps described at the beginning of this section for Netlab, where the number of inputs, outputs, hidden nodes and *hyperparameters* was declared. Then, the network's structure was initialised and some optimising parameters were introduced to control the NN, including the number of training cycles. This step also included some changes in each approach, and these changes will also be described in each section.
5. **NN Training:** In Netlab, this step is also known as network optimisation, and for cross-validation, only the training data points were used to optimise the network. This step provided training error values that could be used to assess the network's performance.
6. **NN Validation:** This validation step consisted in forward-feeding the trained NN using the validation data points, providing error values to assess the network's performance.
7. **NN Testing:** This final step consisted in forward-feeding the testing data points through the same trained NN as the previous step. Again, these steps delivered output error values that gave the final accuracy result of the neural network. All the NN steps from 4 to 7 were the actual cross-validation steps, which were repeated to report the best network's training error.

The selection of the number of hidden nodes for a NN is crucial for its performance and complexity. Therefore, it was important to test different numbers of hidden nodes in each cross-validation, using the different error values to evaluate the best quantity. Additionally, since the NN optimisation includes randomisation of the initial weight values, each optimisation task can provide different results, even whilst using the exact same parameters and data sets. Consequently, it was also decided to repeat each training, validation and testing ten times for each number of hidden nodes, to select the optimal NN. This process is depicted in Figure 7.8, where the tables for training, validation and testing include their respective error values. The manner of selecting the optimal NN was by analysing the validation errors and selecting the optimal one, which had the lowest value with the least complex NN. This decision was performed manually using the author's criteria and will be furtherly described in each approach. Once this optimal NN was selected, the testing error value for that specific NN was the reported value to assess the final accuracy and performance of the NN.

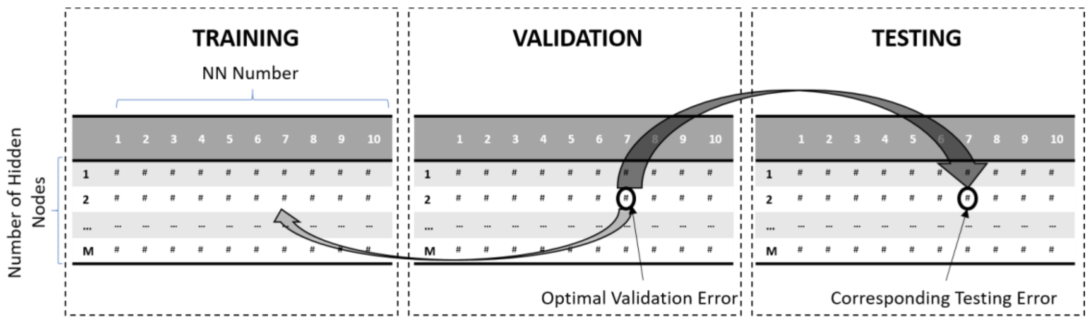


Figure 7.8: Optimal NN selection for different numbers of hidden nodes and neural networks.

7.3.1 Individual Tests Networks

For this first approach, each test was processed separately, so that individual networks were created for each test. Therefore, referring to the NN algorithm presented previously in Figure 7.7, this approach followed the described initial steps of "data loading" and "data pre-processing". However, since the machining parameters in a single test were constant, instead of having six inputs only the spark area and spark intensity were used. After this, during the "data assignment" step of this approach, a variable was created of the same size as input's dimension and with random values between 0 and 1. Then, using a conditional argument in the algorithm, it was possible to separate each test's data points into the training, validation and testing data sets. Therefore, in a single test, all data points that had a corresponding value in the randomised variable between 0 and 0.8 were assigned for training, whilst the ones between 0.8 and 0.9 were used for validation, and between 0.9 and 1 were used for testing. This was therefore in accordance with the mentioned data sectioning of 80% of data points for training, and 10% each for validation and testing.

Furthermore, there exists a difference in opinions regarding the best number of hidden nodes for a NN. For a two-layer NN, the number of hidden nodes dictates the number of weights required. Using Equation 7.10 it is possible to calculate the number of weights in a NN, provided the number of hidden nodes h, the number of inputs i and outputs o.

$$No. of\ Weights = (i+1)h + (h+1)o$$

Equation 7.10

The number of weights of a NN should be significantly lower than the number of input data points. A NN with the same or higher amount of weights as input data points would create an unnecessarily complex, over-trained and overfitted network. Therefore, this “ratio” of a number of weights per number of data points can vary depending on the type of data and neural network. In the case of the present research, three ratios were calculated to understand the results of each tests’ NN, and these were 3x1, 2x1 and 1x1; as can be seen for each test in Table 7.1.

Table 7.1: Number of hidden nodes for different weight-input data points ratios.

TEST NO.	NO. OF NODES RATIO 3 X 1	NO. OF NODES RATIO 2 X 1	NO. OF NODES RATIO 1 X 1	MAX NO. TO EVALUATE
Test 1	7	11	23	30
Test 2	12	18	37	45
Test 3	12	19	39	50
Test 4	8	13	27	35
Test 5	12	28	37	45
Test 6	10	15	31	40
Test 7	7	11	23	30
Test 8	14	22	45	55
Test 9	17	25	52	60
Test 10	10	15	31	40
Test 11	11	17	36	45
Test 12	13	20	42	50
Test 13	8	13	27	35
Test 14	7	11	23	30
Test 15	10	15	31	40
Test 16	8	13	27	35
Test 17	14	21	43	50

The ratio of 1x1, whilst undesired, was considered to correctly evaluate each network to the appropriate number of hidden nodes. The maximum number hidden nodes to evaluate a test was set to a value slightly higher than the 1x1 number, as shown in the fourth column of Table 7.1. Therefore, for instance, in test number 9 a good number of hidden nodes with the 3x1 ratio would be 17, whilst the test was evaluated up to 60 hidden nodes. It is relevant to remember that, since different machining parameters were used, each test has a different number of data points corresponding to their different machining times. It is also important to mention that, since there are two factors in the algorithm that depends on randomisation, namely the randomised variable and the actual NN, each random generator seed was stored to preserve repeatability.

Once the maximum number of hidden nodes to evaluate a test was calculated, the rest of the algorithm would be implemented. Using the selection process described in Figure 7.8, the optimal NN was selected for each test. It was mentioned then that this task was performed manually, where the method of selecting the NN started by finding the minimum values of error with the lowest number of hidden nodes. However, in some occasions, these minimum values would include a number of hidden nodes closer to the 2x1 or even the 1x1 ratio. Regardless of this, the difference in the error value between these complex networks and networks with a low number of hidden nodes was generally quite small. This is where the author's discretion would be used to select the most appropriate NN, finding a balance between the least complex and most accurate one. Therefore, Table 7.2 shows these optimal networks selected, with the number of neural network, number of hidden nodes and final testing error for each test.

Table 7.2: Optimal networks, given by the NN's number and a number of hidden nodes, with final testing error values.

TEST NO.	NN NUMBER	NO. OF HIDDEN NODES	FINAL TESTING ERROR
Test 1	2	7	0.8063
Test 2	9	13	0.455
Test 3	6	11	0.3241
Test 4	6	7	0.2808
Test 5	1	8	0.3386
Test 6	9	12	1.2134
Test 7	9	7	0.1630
Test 8	6	14	0.3640
Test 9	2	3	0.1278
Test 10	10	8	0.0665
Test 11	1	8	0.0559
Test 12	2	7	0.1818
Test 13	10	7	0.0849
Test 14	9	6	0.1896
Test 15	3	10	0.3098
Test 16	1	5	0.5662
Test 17	1	10	0.1954

As can be seen in the table, in most tests the optimal NN was achieved with a number of hidden nodes equal or lower than the 3x1 ratio, with the exception of tests 2 and 6. On the other hand, it was interesting to find that test 9 had the lowest number of hidden nodes, regardless of having the highest number of input data points.

Taking test 2 as a representative example, Figure 7.9 shows the training mean-squared errors for each number of hidden nodes, where the line shows the mean values of these errors. It can be appreciated that the training errors behaved as expected in a NN, as the error decreases rapidly until about 0.27, where the trend stabilises around the error value of 0.25.

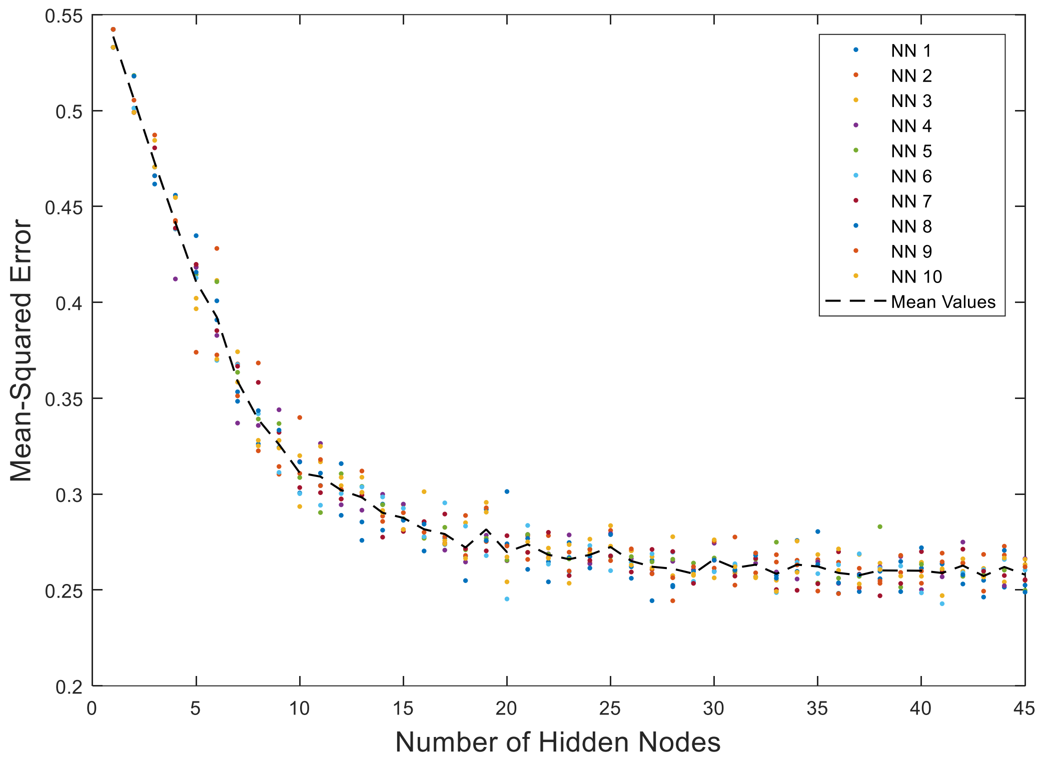


Figure 7.9: Test 2 training mean-squared errors for different numbers of hidden nodes.

Similarly, Figure 7.10 shows the validation mean-squared errors for each number of hidden nodes. Here again, the general trend shown by the mean value line had the expected behaviour, with a rapid descend, followed by a stabilisation, and then a mild increase. As shown in Table 7.2, the chosen optimal neural network for test 2 was the 9th network with 13 hidden nodes, delivering a validation error of 0.2766 and a final testing MSE value of 0.4555.

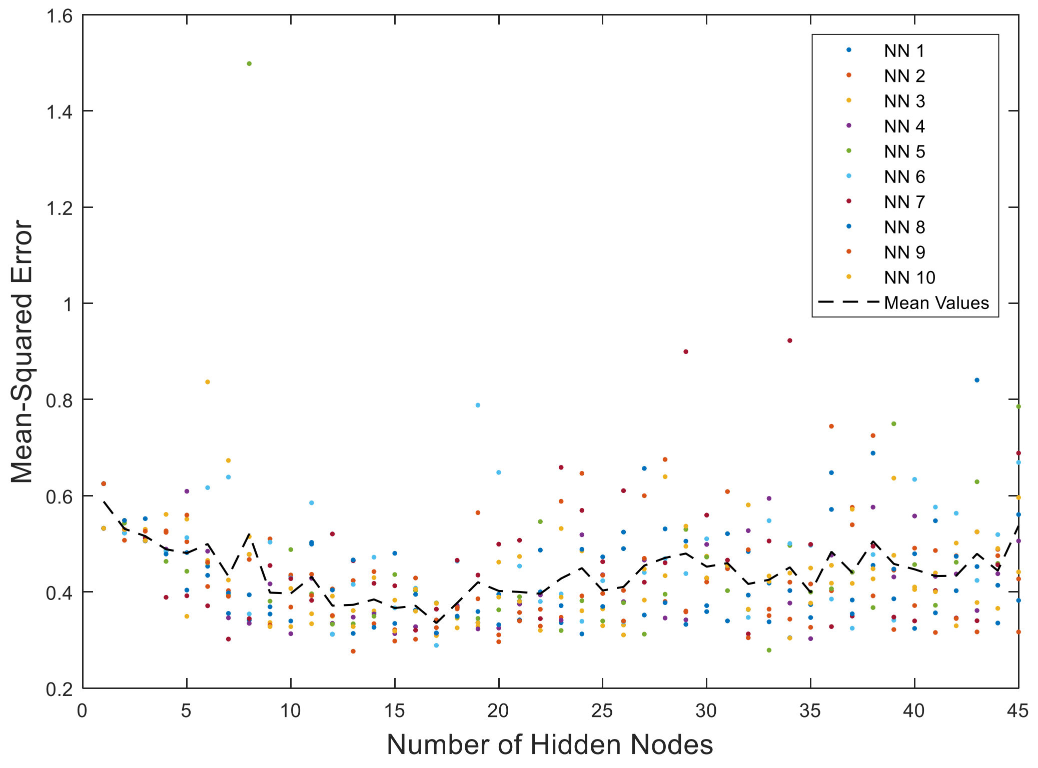


Figure 7.10: Test 2 validation mean-squared errors for different numbers of hidden nodes.

The number of iterations or training cycles was declared as 1,000, and the selection of data points for the cross-validation was divided in 80%, 10% and 10%. However, it may be argued that if these quantities were changed, there would be important changes in the accuracy results of the selected NN. Nevertheless, the actual changes that these factors provided were not very dramatic and arguably not worth the time expense of investigation. For the present case of test 2, if the number of iterations was increased to 10,000, the final training error would decrease to 0.4528. However, in the author's opinion, the improvement in error of less than 0.003 did not justify the increase of processing time by 10 times. Furthermore, whilst in this test's case, the training Bayesian error appears to continue to descend after 1,000 iterations, there were tests where the continuous flat slope would be obtained below this number. Therefore, it would also impact in general pre-processing and processing time of the networks, the labour of deciding which tests would benefit from more iterations.

Similarly, by altering the cross-validation sections for test 2 into 70% training, 20% validation and 10% testing, the validation MSE would rise to 0.5227, even though the number of validation points was increased, whilst the final testing error would be 0.6564. Furthermore, whilst this change delivered poorer results, the difference in testing error was not too great.

Yet, this test could aid in supporting that the initially chosen separation of data points was advantageous.

Going back to the results obtained with the optimal NN, at closer inspection there were interesting factors found. In Figure 7.11 the individual validation errors of the selected NN number 9 appear to be quite disperse, showing the same initial rapid descend expected in validation errors, but followed by constant sharp fluctuations.

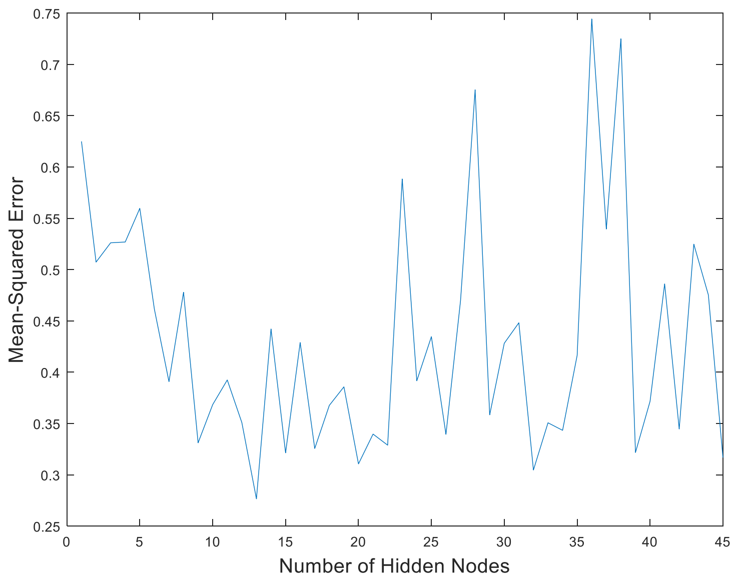
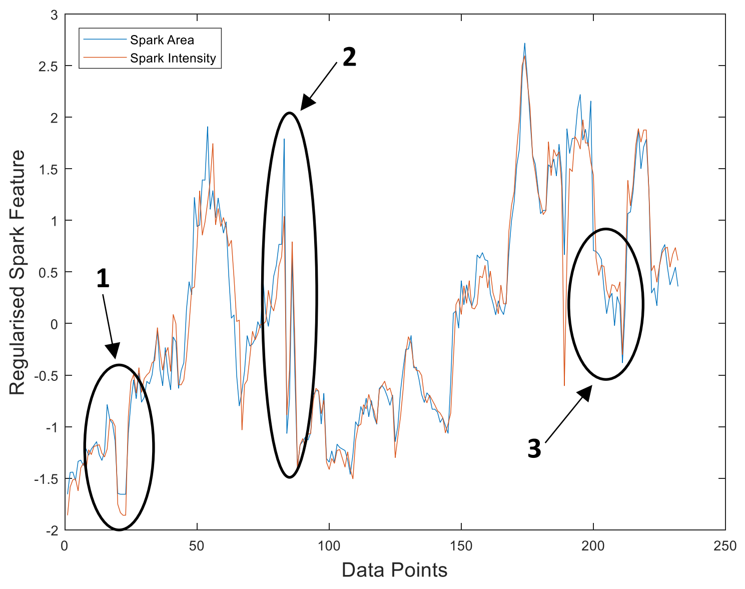
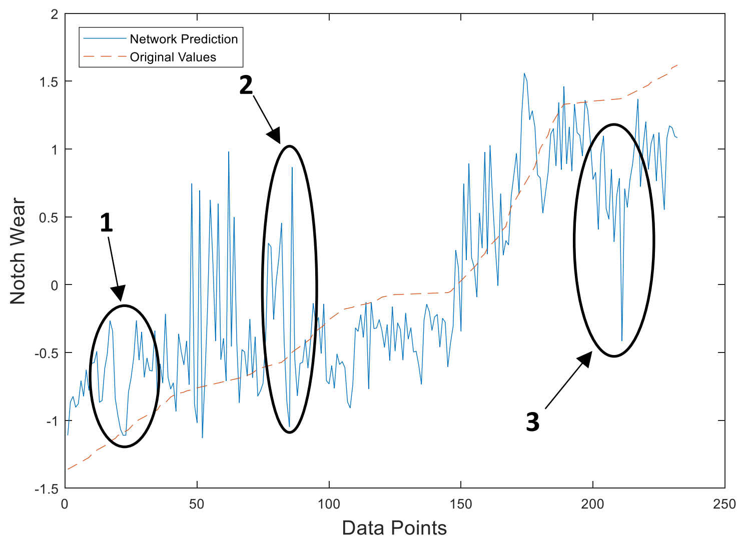


Figure 7.11: Test 2 validation MSE for different number of hidden nodes in NN 9.

Figure 7.12 shows the input values of spark area and intensity in (a), and the regularised training notch wear target and predicted values in (b). It can be seen in (b) that the prediction outline has a similar overall trend as the target values, however, there are plenty of leaps and sudden decreases. In closer inspection, it was found that these leaps and decreases matched the spark behaviour's leaps and decreases. This can be evidenced by the encircled areas 1, 2 and 3, around data points 23, 88 and 211, where the input's behaviour in (a) is very similar to the prediction outline in (b).



(a)



(b)

Figure 7.12: Test 2 input values (a), and training notch wear original target values and network prediction (b).

This was the same situation for the other two output variables of flank wear and notch wear in Figure 7.13, where again the general trend of both appears similar, but with a lot of fluctuations at the same data points as the input values.

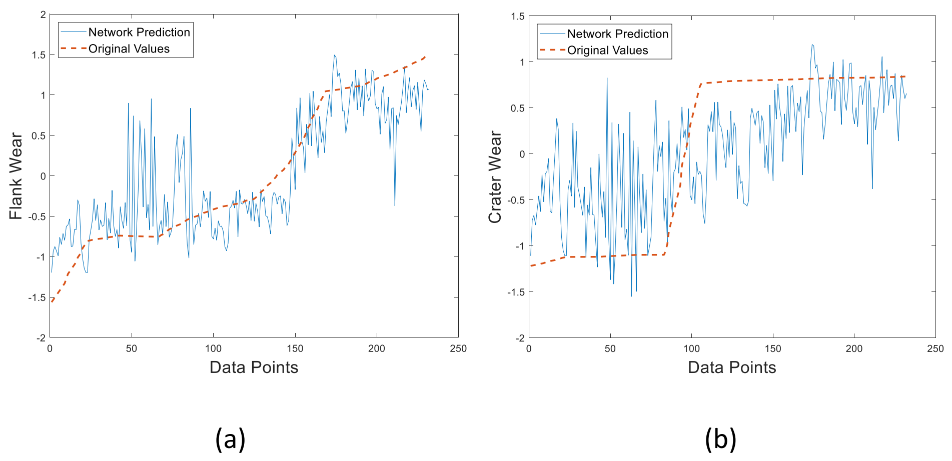


Figure 7.13: Test 2 original training output values and network prediction for flank (a) and crater wear (b).

It is relevant to remember that, even though some data filtering was performed to eliminate the effect of tool engagements and disengagements, some of these remained noticeable in the spark outline. Additionally, there were the “flaking events” described in section 6.3.2, where the spark trend experimented a sudden decrease of area and intensity due to the newly exposed edge of the cutting tool. Therefore, whilst the small noise present in the inputs’ signal appears to be absorbed by the NN, it was found that these larger changes may be confusing the NN. This effect can be explained by Figure 7.14, which on the left figure, one of the spark features (area or intensity) is represented, and on the right, its respective wear feature (flank, notch or crater wear) is also shown. On the spark feature, points 1 and 2 appear to have the same value of the spark feature, $S_1 = S_2$, even though they have different time values, $T_1 < T_2$. On its respective wear feature, on the other hand, the wear values are different, as the wear of the tool is always increasing, $W_1 < W_2$. Therefore, it was found that the network appears to be attempting to assign two different output values to the same inputs.

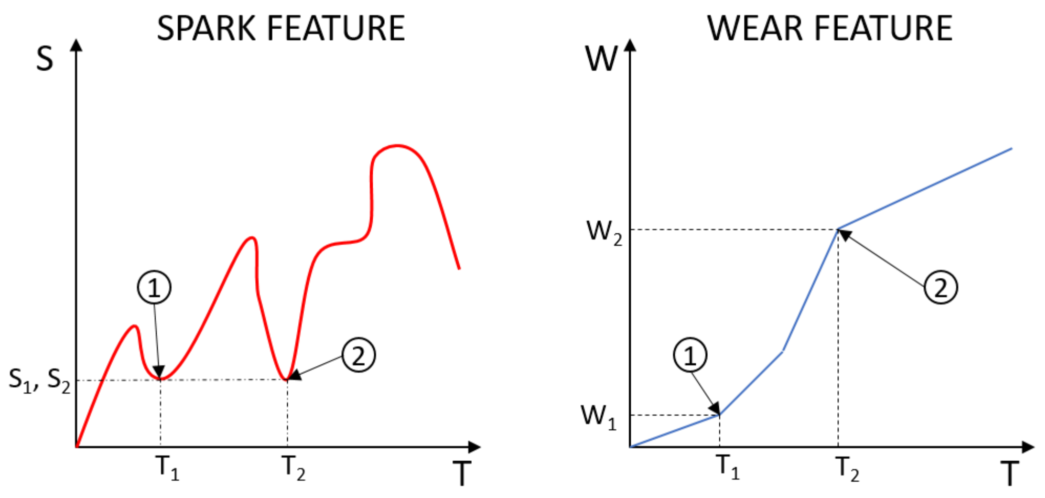


Figure 7.14: Representation of network behaviour regarding a spark feature (left) and a wear feature (right).

It could be a common misconception that neural networks are complicated black box models that do not have limitations, or that can overcome almost any type of difficulty with input data. However, this issue provided a good idea of how this NN is behaving, as well as the causes for such predictions. Therefore, the neural network appeared to be affected by these apparent “confusing variations” during training, and it seemed to be attempting to average the predicted values for every recurrent input value.

All tests had a great deal of fluctuation in their outline, as disengagements and flaking events were present in all of them. However, not every test had these “confusing variations,” where the spark feature would decrease frequently to previous input values. Test 2 and test 6, as mentioned previously, where the tests that required a larger number of hidden nodes that the minimum for a ratio of 3x1. For test 2, it could be seen before that it contained many confusing variations, and after a similar analysis, test 6 also appeared to contain many of these. On the other hand, tests 9, 10 and 11 with low MSEs and low NN complexity, whilst also presenting many input fluctuations, they appeared to contain less confusing variations, as shown in (a), (b) and (c). Therefore, it was found that the complexity of the NN and the final testing value of MSE appeared to be correlated to the number of confusing variations in a test.

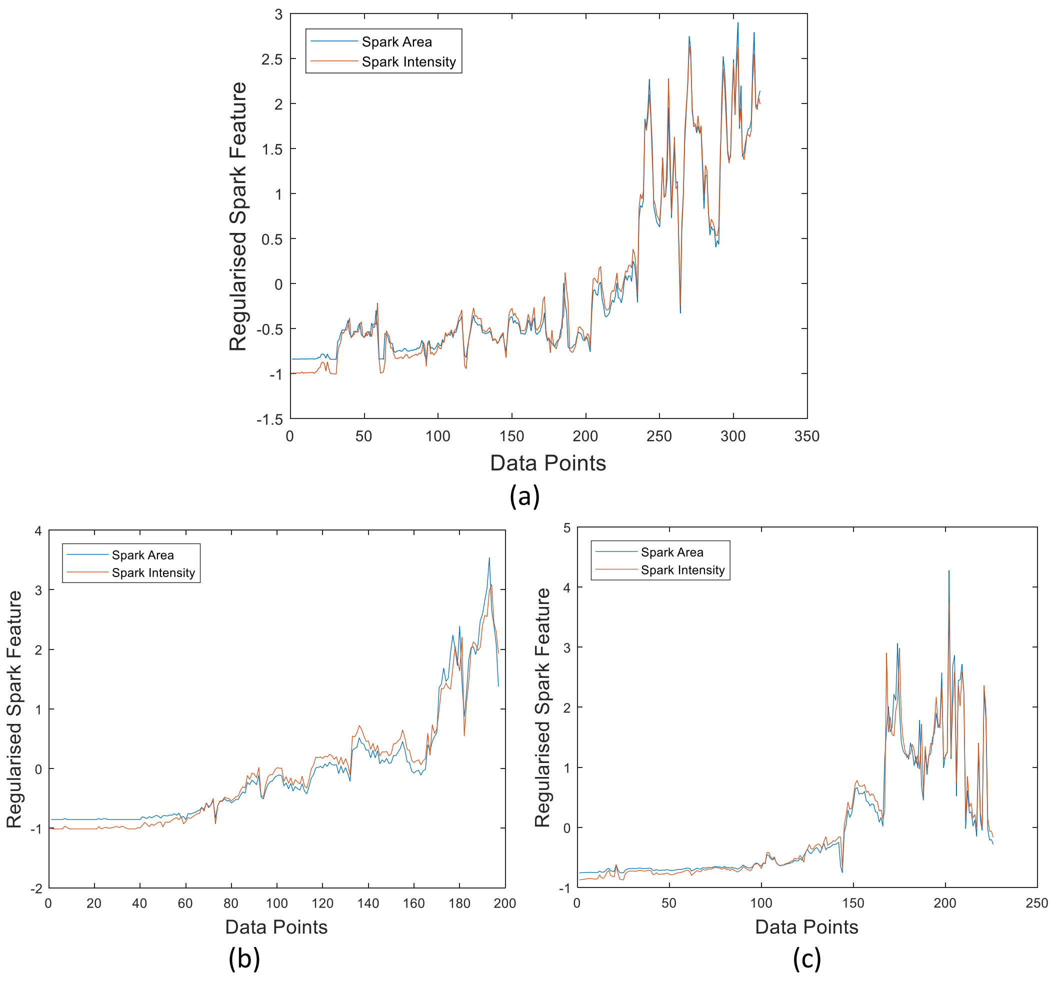


Figure 7.15: Training inputs of spark area and intensity in test 9 (a), test 10 (b) and test 11 (c).

After careful analysis, it became clear that the main difficulty relayed in finding a way to inform the NN that a new input that had the exact same value as a previous one, was occurring at another moment of the machining task. Therefore, a possible solution found for this problem, caused mainly by disengagements and flaking events, was the introduction of a new input. This would allow the existence of a three-dimensional input array, where this new value would allow the NN to change the plane of input values when a disengagement or flaking event was found. This is represented by Figure 7.16, where S_A is the spark area and S_I is the spark intensity. There, when the input level arrives to point (1), where a disengagement or flaking event occurs, the new "spark event" input S_E would switch to another plane in point (2). This way, the decrease in the spark features to point (3) would occur without overlapping previous values at the initial plane.

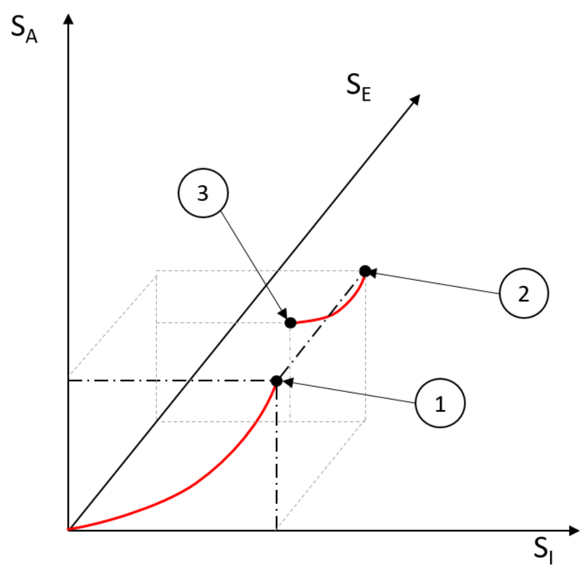


Figure 7.16: Representation of the introduction of a spark event S_E input variable.

Therefore, this new approach would require the addition of a step at the pre-processing stage of the NN algorithm, where this spark event was calculated. As was mentioned in section 6.3.2, flaking events generally included a sudden decrease in the spark evolution, where the lower level of spark area and intensity was maintained for a few seconds. After careful analysis, it was found that five seconds was generally the minimum number of seconds that the new level of spark was maintained. Therefore, this new step compared five past values of spark features with a buffer of the next five data points. If the buffer points were all lower than a certain percentage of the past five values, the spark event variable would be assigned a 0.01 value. Then, the spark event vector would have this value of 0.01 in subsequent points until the next event was detected, where the value would be increased to 0.02, and so on.

This solution was applied to the representative example of test 2, and the changes in the prediction and error values were dramatic. In Figure 7.17 the training predicted outputs and target values of notch, flank and crater wear are shown. It can be seen how the correlation between the trends was importantly improved in the three outputs, and the testing MSE was reduced by more than 90%. This can be appreciated in Table 7.3, where the number of hidden nodes and the resulting testing errors of the previous method without the spark event input are compared with this new approach with the spark event input. In most tests, the number of hidden nodes was reduced achieving less complex networks, and the final accuracy of all test was significantly improved.

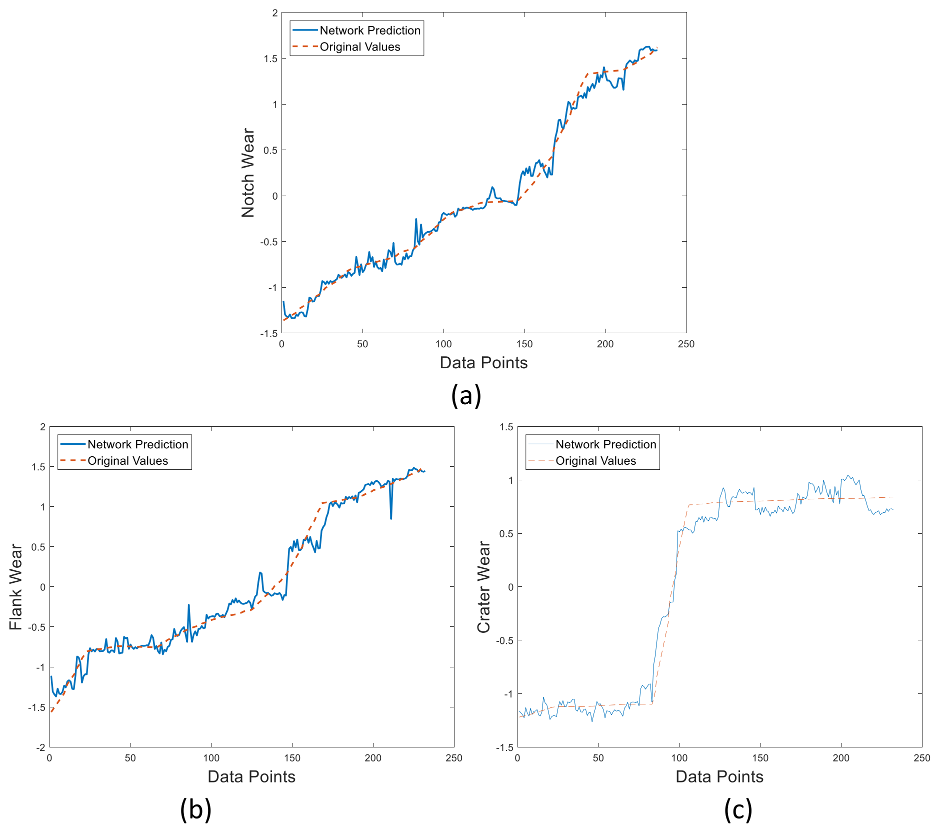


Figure 7.17: Test 2 original training output values and network prediction for notch (a), flank (b) and crater wear (c), using spark event input.

Table 7.3: Compared results with and without the use of the Spark Event input, given by number of hidden nodes and final testing error values.

	RESULTS WITHOUT SPARK EVENT		RESULTS WITH SPARK EVENT	
TEST NO.	NO. OF HIDDEN NODES	FINAL TESTING ERROR	NO. OF HIDDEN NODES	FINAL TESTING ERROR
Test 1	7	0.8063	7	0.0455
Test 2	13	0.455	4	0.0225
Test 3	11	0.3241	9	0.0618
Test 4	7	0.2808	5	0.0418
Test 5	8	0.3386	7	0.0151
Test 6	12	1.2134	6	0.0087
Test 7	7	0.1630	3	0.0789
Test 8	14	0.3640	14	0.0198
Test 9	3	0.1278	3	0.0294
Test 10	8	0.0665	6	0.0392
Test 11	8	0.0559	6	0.0278
Test 12	7	0.1818	9	0.0125
Test 13	7	0.0849	6	0.0356
Test 14	6	0.1896	6	0.0144
Test 15	10	0.3098	6	0.0234
Test 16	5	0.5662	8	0.0179
Test 17	10	0.1954	11	0.0257

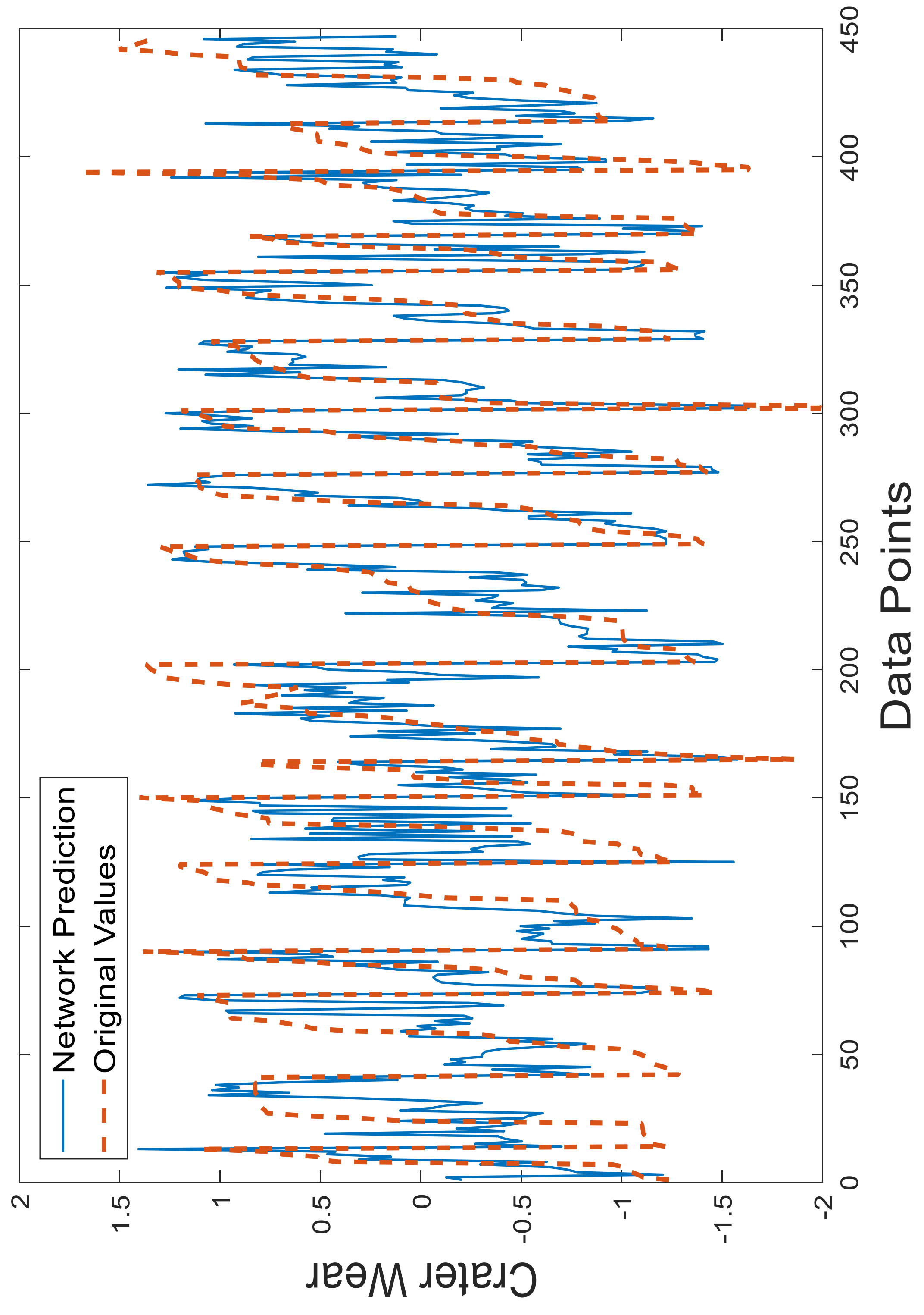
Before the implementation of this new input, there was a concern that, given that the spark event variable would have a less fluctuating and overall increasing trend, the NN would instead attempt to only map that variable for the prediction. However, it could be seen in the previous figures that the general fluctuation of the spark features is still present in the network prediction outlines. Therefore, this approach result was found to be successful and applicable for the next steps of this chapter.

7.3.2 Randomised Data Network

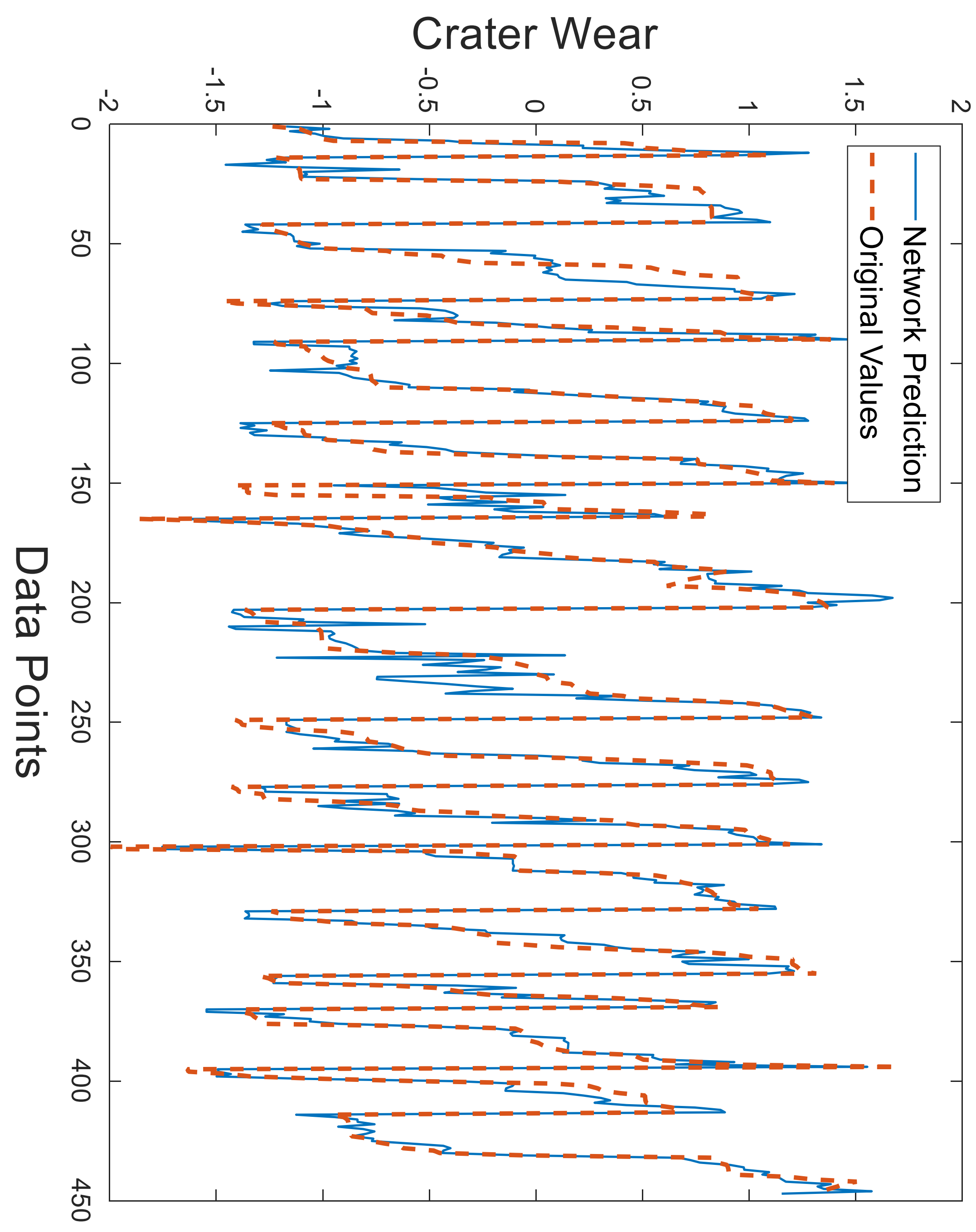
This second approach, as summarised at the beginning of this section, consisted of using all the data points from all the tests. The same approach as in the section 7.3.1 was implemented at the data assignment step of the NN algorithm, where a variable was created to randomly divide all points into the training, validation and testing data sets. Also, at the NN initialisation step of the NN algorithm, the other inputs related to the machining parameters were included. Finally, the same approach of optimal NN selection was used as in section 7.3.1, however, in the present approach, only 5 NN were processed, instead of 10. This was decided due to the processing time of the networks in this approach, where the amount of data points was much larger than in the individual tests networks, and hence, the number of hidden nodes tested was higher. In this approach, the total number of data points of all the tests ascended to 4,428, providing an 80% training data set of around 3,542, and 10% validation and testing data sets each of around 442 data points. With this larger number of training data points, the previous low data to weight ratios of 3x1, 2x1 and 1x1 would mean 118, 177 and 356 hidden nodes each. Therefore, the processing time for 10 networks with 356 hidden nodes was found to require almost a week of processing time.

However, in addition to that reduction of networks and given the large number of data points, it was found that the ratio of input data points per weight could be improved into an 8x1 ratio, where around 44 nodes would be expected. Consequently, it was finally decided to evaluate 5 networks and a maximum number of hidden nodes of 200, in this approach. It is relevant to mention that even this “reduced” number of loops in the NN algorithm took around 25 hours of processing time.

These parameters were implemented, and optimal networks were evaluated, both with the initial 6 input variables (spark area and intensity, cutting speed, feed, cutting depth and radial immersion) and with the inclusion of the spark event variable. The resulting validation predictions and the original target values can be found in Figure 7.18, where (a) was obtained with the original 6 inputs and (b) with the inclusion of spark events.


Network Prediction
Original Values
Crater Wear
Data Points
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
50
100
150
200
250
300
350
400
450

(a)



(b)

Figure 7.18: Crater wear validation target values and network prediction for 6 inputs (a) and 7 inputs (b)

The optimal neural network selected for the first had 27 hidden nodes and a testing MSE of 0.3025, whilst the inclusion of spark event approach reduced the number of hidden nodes to 25 with a testing MSE of 0.0673. Here again, the spark event input aided in the general performance of the NN, improving its accuracy by about 77%. Yet, there is still much fluctuation in the prediction outline as the disengagements and flaking events still impact in the NN performance.

These results showed a successful NN performance, validating the use of the machining parameters and the spark event as inputs of the NN. However, the manner of selecting points for the cross-validation greatly aided the NN performance, as the interpolation was more direct, having data points from all tests at training. Therefore, for the next step, it was decided to use entire tests to get results of a more demanding scenario.

7.3.3 Full Test Data Network

This final step consisted of using the entire data points of a number of tests for cross-validation. For this reason, at the data assignment step of the general NN algorithm, the use of orthogonal arrays at the DOE helped in sorting the tests into the different data sets. As it was mentioned in section 3.3.3.1, the full orthogonal array with the 18 tests could be divided into two arrays of 9, previously shown in Table 3.10 and Table 3.11 in that section. However, in the second array, the 15th test had to be omitted due to the error in the video feed described in this chapter's section 7.1, making a total of 17 tests. Therefore, given the mentioned assignment of 80%, 10% and 10% for cross-validation, these tests were divided into 13 tests for training, 2 for validation and 2 for testing. Furthermore, as orthogonal designs have the property of providing pairs of combinations in experiments, the entire first array along with four more tests of the second array, were assigned to training, as shown in Table 7.4. The remaining four tests of the second array were sorted into the validation and testing data sets shown in Table 7.5 and Table 7.6 respectively.

Table 7.4: Tests for training in final network's cross-validation.

TEST NUMBER	CUTTING SPEED Vc (m/min)	FEED fz (mm)	CUTTING DEPTH ap (mm)	RADIAL IMMERSION (%)
8	700	0.07	1	50
9	700	0.07	1.5	90
2	700	0.09	1	70
3	700	0.09	2	90
10	700	0.11	1.5	50
12	850	0.07	1.5	70
18	850	0.07	2	50
16	850	0.09	1.5	90
4	850	0.11	2	70
5	1000	0.07	2	90
13	1000	0.09	2	50
7	1000	0.11	1	90
14	1000	0.11	1.5	50

Table 7.5: Test for validation in final network's cross-validation.

TEST NUMBER	CUTTING SPEED Vc (m/min)	FEED fz (mm)	CUTTING DEPTH ap (mm)	RADIAL IMMERSION (%)
17	850	0.11	1	90
11	1000	0.07	1	70

Table 7.6: Tests for testing in final network's cross-validation.

TEST NUMBER	CUTTING SPEED Vc (m/min)	FEED fz (mm)	CUTTING DEPTH ap (mm)	RADIAL IMMERSION (%)
6	850	0.09	1	50
1	1000	0.09	1.5	70

Furthermore, the spark event input was included at the NN initialisation of the general algorithm, as it was previously demonstrated that this input improved the NN accuracy. Given that this would be the final NN approach, it was decided to build again 10 networks, each evaluated to a maximum of 200 hidden nodes. The processing time of this NN was extremely long, as it took several days to finalise, however, it was deemed relevant to improve the network's possibility of providing the best results.

The neural network selected was the 9th network with only 6 hidden nodes, and its validation predictions against target values are shown in Figure 7.19, Figure 7.20 and Figure 7.21 for each wear output. It can be seen how this final approach required a much lower number of hidden nodes to obtain a low value of validation MSE, and the trends of the target values and network predictions are again quite similar. However, the resulting testing MSE was of 1.3792, and its predictions against target values can be seen in Figure 7.22, Figure 7.23 and Figure 7.24. In these images, it is possible to see that the trends are again quite similar, but growing in parallel with a gap between each other.

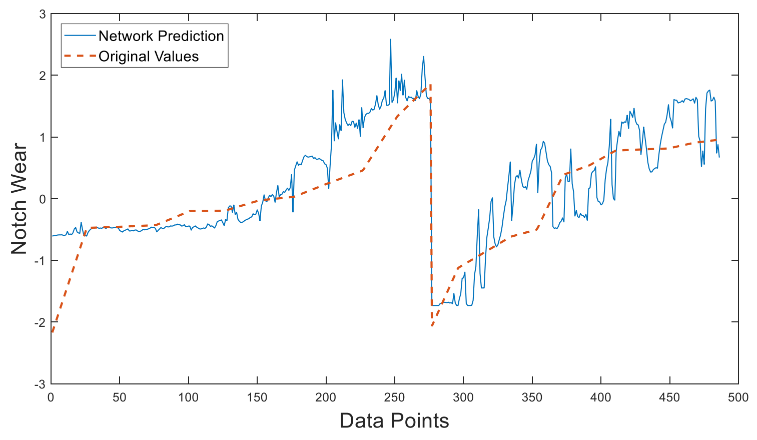


Figure 7.19: Notch wear validation target values and network prediction for full tests network.

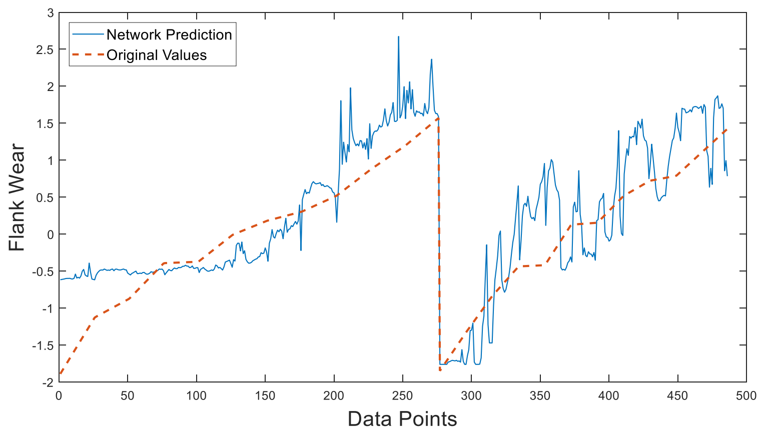


Figure 7.20: Flank wear validation target values and network prediction for full tests network.

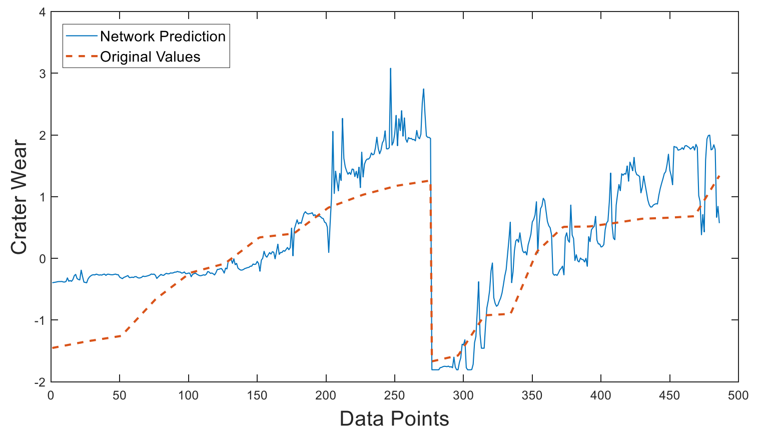


Figure 7.21: Crater wear validation target values and network prediction for full tests network.

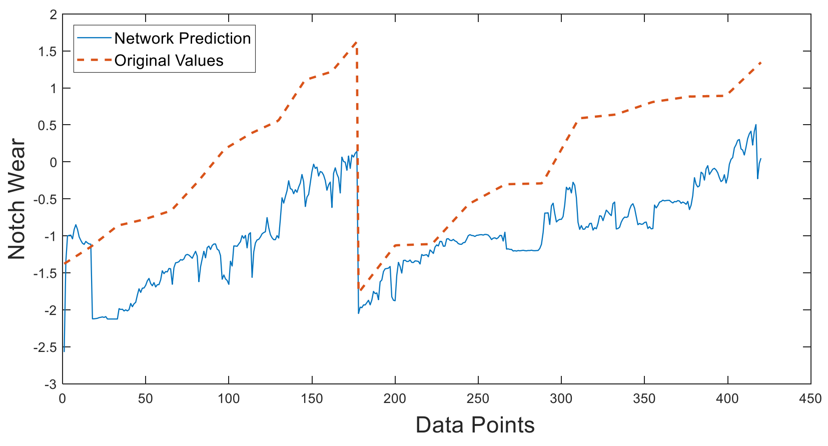


Figure 7.22: Notch wear testing target values and network prediction for full tests network.

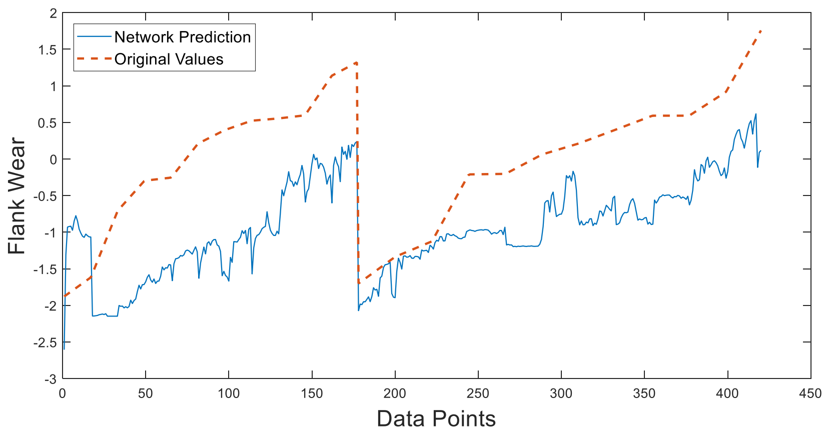


Figure 7.23: Flank wear testing target values and network prediction for full tests network.

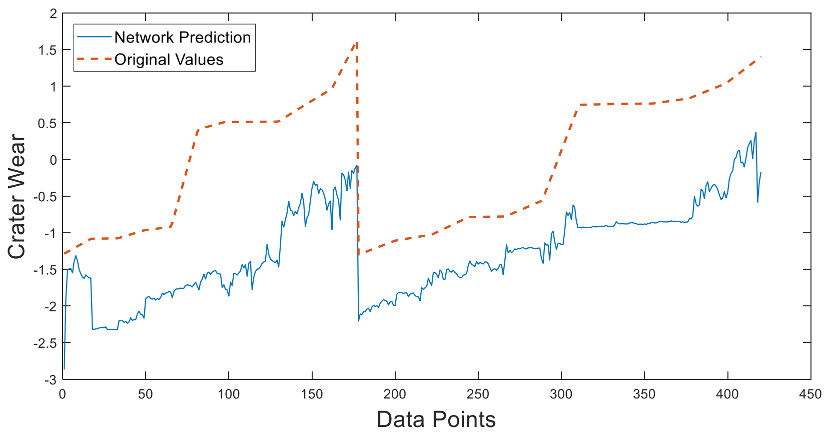


Figure 7.24: Crater wear testing target values and network prediction for full tests network.

This approach included a more demanding scenario, as the validation and testing data sets included entirely unseen states of spark evolution and new combinations of machining parameters. The performance of the NN was successful in approximating the known wear values at validation, however, the resulting testing MSE was quite high. Nevertheless, the testing data set included test 1 and 6, which were some of the most challenging tests regarding spark events and input fluctuation. It could be argued that by selecting other combinations the NN could provide more accurate results. However, it is almost certain that the improvement would not be very high. Additionally, in an industrial scenario, which would be the final objective of this research, it would be desired to have a robust NN that could overcome such difficulties. Furthermore, the possibility of carrying out all the possible combinations to test these arguments would have exceeded this research's timeframe due to processing time.

The data acquisition methodology for the research should also be taken into consideration when assessing these results, as the NN performance is also related to decisions made then. The constant interruption of machining created the disengagements, which is one of the main reasons of the undesired fluctuation of inputs. Also, the general illumination problem described at the testing parameters section, whilst tackled by the revisions mentioned to the algorithm methodology, they may be furtherly explored to achieve less variable spark results.

However, the main reason for the result obtained in this last approach was more related to the quantity of data and the spectrum of testing examples. Therefore, it is relevant to remember that all this analysis was based on the orthogonal arrays of the DOE, as this method was selected to minimise the number of tests whilst assuring that all pairs of combinations were tested in each 9 tests array. This property had the apparent advantage of providing enough combinations for the training of the NN by including an entire array plus four more examples from the second array. However, it was found that these designs did not provide enough tests data to successfully populate a suitable cloud of examples for the window of machining parameters chosen and their respective combinations. Instead, these examples seemed to provide isolated and punctual cases for training, forcing the NN to extrapolate at the validation and testing steps. Neural networks are excellent in interpolating, as they are very good approximators of input functions where enough examples are provided during training. However, they can be quite unsuccessful in extrapolation, given the nature of their design and optimisation. This explains why the behaviour of the NN was apparently less accurate in this final approach.

Therefore, more tests would have been necessary to achieve more accurate results in this last scenario, where the entire range of machining parameters should have been tested and used for training. Also, it would have been desirable to carry out some tests with the same parameters, to further understand the repeatability of the ceramic behaviour. Nonetheless, this study was performed in the final stages of the author's doctoral research, and the timeframe was limited, preventing more material acquisition, experimental planning, and in general, more testing. Yet, the results and conclusions described in this chapter were very interesting and representative of this approach's capabilities and limitations.

7.4 Summary

In the current chapter, a two-layer neural network structure was implemented to predict notch, flank and crater wear; including initially six inputs, namely spark area, spark intensity, cutting speed, feed, cutting depth and radial immersion. Also, a general NN algorithm was presented, which included the steps that were implemented in all the networks created.

To fully understand and analyse the behaviour of the NN with the data that was acquired during the third set of experiments, three approaches were developed. In the first approach, each test was used individually to build and select an optimal NN through cross-validation. It was found that the created networks, whilst showing a correct general performance, they were being confused by the very variable inputs' outline. These inputs included many spark feature fluctuations due to tool disengagements and flaking events. Therefore, to overcome this issue, a new input named spark event was introduced. This new input was created by processing the spark features' data information so that disengagements and flaking events could be identified and used to extend the inputs' dimensional distribution. Through this spark event input, it was possible to improve the individual tests networks' accuracy by a 90%.

The second approach, on the other hand, included all the data points from all the 17 tests of the third set of experiments. The data points were randomly sorted between the cross-validation data sets, and the resulting testing accuracy using the original 6 inputs was compared to the testing accuracy of the inclusion of the spark event input. The results showed that the neural networks appeared to be delivering a general adequate performance, and again the NN's accuracy was improved by around 77% in the second instance, proving that this input was successfully improving the networks.

The final approach consisted in using the entire data points of the tests, dividing them into the cross-validation data sets using the orthogonal designs of the DOE. Furthermore, the spark event input was directly included and an optimal NN was selected through the cross-validation. However, the final performance of this NN was less accurate than the previous two steps, as the final testing error was higher. However, this lower accuracy was mainly caused by the low number of tests and data points during the training of the NN. The selected DOE had, at first glance, the apparent capability of providing an appropriate range of combinations for training, enabling the desired interpolation of data points. Nevertheless, the number of examples obtained through the DOE was not enough, and at the validation and testing steps, the NN was forced to extrapolate.

Yet, the final NN's prediction trend was quite close to the target's outline, and the final value of error was quite low, considering all the difficulties previously mentioned. Therefore, it could be argued that if the number of test examples was extended to a suitable amount, and the illumination and disengagements difficulties were also tackled, a much more accurate model of tool wear prediction could be achieved.

8 GENERAL DISCUSSION AND CONCLUSIONS

Even though there were some discussion and conclusions included in each chapter of this thesis, this chapter will gather all these and provide some insights into the author's views on the different findings.

8.1 Conclusions

This research was based around the six main objectives presented in section 1.2. Therefore, this section will include the conclusions drawn against each one of these objectives.

1. The first objective consisted on reviewing the literature available on Tool Condition Monitoring (TCM), with a special focus on the research involving vision-based systems. Section 2.2 included this literature review on TCM, with a section (2.2.1) dedicated to vision TCM systems. In these sections, the foundations and main theoretical bases of TCM were described, along with a summary and critical analysis of the different literature on vision-based systems. There, the importance of TCM in the machining industry was laid out, showing that there is plenty of research work on this area and on the different techniques to monitor tool wear. It was concluded that vision-systems of TCM constitute a relevant area of research, but that it has a much lesser development when compared to other systems and techniques. However, an important conclusion that could be drawn from these sections is that there is a strong increase in the use of pattern recognition and machine learning techniques in TCM.
2. The second objective was to carry out experimental tests to collect machining cutting sparks and tool wear data. Therefore, chapter 3 included the common methodology of the research, describing the different experiments carried out for each chapter's objectives, with the general aim to support the final conclusions. This was evidenced in the evolving nature of the experiment, as the conclusions found in each set helped in planning the subsequent ones. Therefore, in the first experiments, where the author had little control on the apparatus set-up and test planning, the image acquisition approach was quite simplistic, acquiring still pictures at a much-extended sample rate. Next, the second set of experiments attempted to overcome this by increasing the sample rate of image acquisition. Finally, the third set of experiments took all the conclusions from these two initial tests, to create a final set of experiments. This final set of experiments attempted to avoid too many time consuming and highly costly tests by using a DOE with orthogonal designs. It would

have been beneficial to have a higher number of tests, as the unpredictable nature of ceramic wear was corroborated by this research. However, a larger DOE such as a factorial design would have required a large quantity of materials and long machine times, as well as a massive, and perhaps difficult to store, amounts of data. Similarly, it would have been interesting to have acquired a parallel set of data such as acoustic emissions or cutting forces, as these have been used to evaluate wear data in the past, as described in the chapter 2. This could have made possible to expand this study into a mixed monitoring model. Nevertheless, this would have required more time to simultaneously process those factors, creating other challenges and impacting on the storage of data.

3. The third objective consisted on assessing the wear of SiAlON inserts and to analyse and select the appropriate wear measuring technique to be used in the present research. Chapter 4 included the wear measuring techniques used to collect flank wear and crater wear area data. A stereo vision approach to measure wear volume was attempted, and whilst unsuccessful, it provided interesting conclusions. The concept of using regular cameras with a stereo vision analysis could potentially provide a low-cost and in-situ wear assessing technique. However, the capabilities of the system used in the present research were limited and perhaps additional resources and technologies could have aided. At present, there are low-cost and highly accessible systems and technologies that would be worth exploring for this purpose, such as the Microsoft's Kinect or Digital Image Correlation (DIC) techniques. Nevertheless, the low accuracy of these types of systems is a continuously challenging issue, as the dimensions of tool wear are quite small. Therefore, further work in that area could be beneficial to find a way to improve these systems' accuracy or to potentially create a bespoke system using similar technology.

   Regarding the wear assessment of cutting tools, chapter 4 provided evidence that lower levels of total tool wear were found with the highest cutting speeds and with the lowest cutting depths. A comparison with the literature found on SiAlON wear by Tian et al. (2013b), Zheng et al. (2016) and Renz et al. (2015) was included, finding common and contrasting conclusions. Compared with the first author, the severity of crater wear was found to be more closely related to cutting depth instead of cutting speed. Regarding flank wear, that source found that flank wear was more severe at higher speeds, whilst the current research's findings were closer to the conclusions presented by Zheng et al. (2016) who also found higher flank wear severity at low speeds. Furthermore, a correlation between Renz et al. (2015) and their "tribolayers" was found when compared with the research's wear assessment conclusions. These conclusions were that there appears to be a window of optimal parameters for SIAlON cutting tools machining nickel-based super alloys. It is believed that the composition and general brittleness of these ceramics may require certain cutting parameters to achieve specific temperatures and contact conditions to successfully cut the material with a low tool wear. This may be the cause of the increased wear results with higher cutting depths and radial immersions, as the contact area was higher between the tool and the workpiece. However, the main findings of the wear assessment section were the ones related to the flaking and chipping effect found in crater wear. It was identified that this mechanism had a very particular and relevant development, which was furtherly explored and used in chapter 7.

4. The fourth objective consisted on evaluating and selecting the optimal image acquisition systems and parameters for the application described in this research. To carry out this analysis, chapter 5 used the first two experiments described in sections 3.3.1 and 3.3.2. The first experiments included datasets from fast, medium and slow image acquisition speeds. After extracting and analysing intensity as a spark descriptor, it was shown and concluded that slow speed images delivered the most suitable representation of the spark for this research. Slow acquisition settings captured a more solid and consistent spark, which displayed a clearer evolution. Faster speeds, on the other side, tended to capture instantaneous conditions of the spark, which created a more randomised spark evolution. However, the low sample rate used in the first tests was identified to be limiting the overall vision of the spark behaviour evolution. In the second experiments, a video feed was captured using a high-speed camera to tackle the first tests' limitation and to be able to test a wider spectrum of possible sample rates or acquisition speeds. Through those tests, the processing of very high sample rates proved again that these were not suitable. On the other hand, an image combination algorithm was implemented to emulate low speeds, which delivered more flexible, richer and adequate results. Therefore, it was concluded that a continuous image acquisition system consisting of a video feed was better than using still pictures. Furthermore, the use of lower sample rates was found to be more adequate, whilst the introduction of an image combination task successfully emulated the longer exposure found in slow image acquisition settings.

There, a video feed with an SLR camera was used, to have a rich data set with high levels of acquisition control. Also, the image combination was implemented to process the video frames, emulating the slow settings, as the combinations were made according to the video framerate. A combination of all the 50 frames in a second of video was selected, which was found to deliver excellent results, as the tool speed was considerably faster at around 5000 rpm.

5. The fifth objective, regarding the evaluation and confirmation of a spark-tool wear relationship, was undertaken in chapter 6, using again the first and second experiments. Through the results obtained by processing the spark area and intensity evolution in the first tests, it was possible to find a very close qualitative correlation with wear. Both trends appeared to be increasing through time, with a similar slope. However, the low sample rate used in the first tests delivered low-resolution approach for closer assessment. Therefore, the second experiments, provided a more detailed level of information, which showed the existence of a phenomenon described in this research as "flaking events". These events consisted of a loss of material from the cutting tool's rake face that created a dramatic increase in crater wear and mass loss whilst showing a sudden drop in spark evolution. This flaking effect appeared to be creating a newly sharpened edge of the cutting tool, lowering the spark evolution, but weakening the insert. Even though flank and notch wear mechanisms created poorer surface finish and accuracy, they did not seemed to be weakening the ceramic inserts as much as the crater wear. Therefore, crater wear, along with its flaking mechanisms, was found to be the main mechanisms of tool failure. Consequently, when the spark area and intensity descriptors were compared to crater wear and mass loss, again a close qualitative correlation was found. These flaking events were visibly identifiable in the spark evolution trend, and they successfully described some of the spark feature changes found throughout a machining operation. Thus, it was concluded that both tests successfully confirmed the research hypothesis, finding evidence of a spark – tool wear relationship.
6. Finally, the sixth objective included the implementation of a machine learning algorithm to predict tool life. Chapter 7 included the use of neural networks to attempt to predict tool wear values, given certain spark information and machining parameters. Three approaches were used to understand and test the accuracy of the created neural networks. The first approach consisted of randomly dividing data points from each test into training, validation and testing data sets; generating and

selecting a neural network per test. The second approach consisted of the same random distribution of data points, but from all the tests into a single neural network. Finally, the third approach divided entire tests sets into the cross-validation data sets to create a single neural network, fed with completely new data at the testing phase. The general performance of the networks in all these approaches showed that the selected toolbox and algorithms were performing successfully. However, in the first approach it was found that the NN was attempting to map two different output values of wear to recurrent similar spark input values; this was because the spark area and intensity were constantly reduced during tool disengagements and flaking mechanisms. Therefore, the implementation of a new input denominated spark event was successful in mitigating this problem. The improvement in accuracy was quite considerable in the first and second approach. The final approach, however, had a more demanding set-up, as the NN was presented with completely unseen tests at the validation and testing sections of the cross-validation. In that final step, the testing error was noticeably higher than in the previous approaches, yet there were specific reasons for that lower NN accuracy. The DOE was used to divide the tests into the cross-validation data sets, relying on its property of generating a good coverage of data. Nevertheless, the performance of the NN suggested that the tests included very isolated conditions of machining parameters that did not manage to provide enough input information to the NN. Therefore, the NN was forced to perform extrapolation with both, the validation and testing data sets. It was mentioned in chapter 7 that neural networks are capable of interpolation if the training data has a good coverage of the input data points. This is the reason for the successful performance of networks in the first and second approaches, as the interpolation was done inside the cloud of data points. Regardless, the final NN achieved a quite low level of error considering these limitations, and the general trends of the predicted and target values were quite similar.

Therefore, in the author's opinion and considering the data acquired, the experience at experimental testing and the analysis of the behaviour of the network, it is believed that with further work, a successful system could indeed be achieved. To accomplish this, more testing would definitively be necessary to get a wider range of examples for the training of the NN. Perhaps another type of DOE would be beneficial to get more combinations of testing examples, and in the author's opinion, repeated tests of similar machining properties would be useful to better understand the ceramic's wear behaviour and to test the NN capabilities.

It can be concluded that the novelty of this research resides mainly in the use of an unconventional machining signal output, such as the cutting sparks, to monitor a cutting process. This could lead to an industrial uptake in the use of image processing and computer vision to analyse different manufacturing process, using visual information and conditions that may have been overlooked before. Furthermore, in the case of the spark-tool wear technology, future advancement in TRL could generate an uptake in its implementation and use in all the machining process where it may be applicable.

In the research side of this work, a publication could be generated to present the research hypothesis and the conclusions shown in chapter 6. Another publication could include the selection of the image acquisition systems and parameters. And a third publication could include the predictive algorithms described in chapter 7, which show the capabilities of this technology and the need of further research.

8.2 Contributions

The main contribution that this research attempted to provide was the basis of a spark – tool wear relationship for high-speed machining of nickel-based super alloys, using SiAlON cutting tools. In the previous chapters, evidence of this hypothesis was presented, and the existence of such a relationship was proved. This relationship is the pillar of the proposed Tool Condition Monitoring (TCM) system that could potentially perform a live assessment of tool wear.

Inside the mentioned main contribution, several aspects of this TCM system were established for possible further work. This was the case of the optimal image acquisition systems and parameters, where it was concluded that a video feed and an image combination algorithm as an emulator of slow acquisition settings, where the optimal options for this approach. Similarly, the main variables that were important to assess the evolution of cutting sparks and its relationship with tool wear were established. The spark area and intensity extraction approaches were successful in providing a constant sensing of spark evolution, whilst the flank, notch and crater wear mechanisms were found to be the most relevant for comparison. And finally, the approach on tool wear prediction using neural networks provided a good idea of the type of challenges and capabilities of this pattern recognition method could encounter in this approach. Most importantly could be the challenge found regarding the confusing behaviour for the NN of the spark inputs, and the suggested solution of an event counter, which enables the expansion of the input dimensional array.

Parallel to these, other general contributions were made around the present research. One of them was regarding wear measurement of cutting tools, where a model of crater wear area was described. Other researchers have developed similar wear measurement techniques through tool imaging, however, the one presented in this research was in the author's opinion, more simplistic and quite accurate. Additionally, the wear assessment of cutting tools using stereo vision was a novel approach that, whilst unsuccessful in the present research, could be furtherly explored to create a low-cost and effective in-situ tool measuring technique. Another parallel contribution was made regarding the results and conclusions of the wear assessment of SiAlON materials, where interesting results were found and contrasted to the limited existing literature. This assessment could have an important input to the area of tool wear assessment of SiAlON ceramics, as it is the author's opinion that more research on this area is required, as these materials are widely used. Yet, the main contribution of this assessment would be related to the found relationship between tool flaking events, crater wear and the microcracks. Even though a detailed analysis of the microcracks was not carried out, their existence and relationship with the other two could be crucial to potentially unify a tool failure model. Furthermore, these findings led to the novel implementation of the spark event detector at the NN, which had a huge effect on the results obtained. This spark event detector was used to detect flaking events, providing the networks with a dimensional expanding solution, given the behaviour of spark evolution in these machining operations. This solution consequently led to a significantly improved accuracy in the networks created.

8.3 Further Research Implications

It is believed that this research has plenty of further research, as there is much potential in this type of vision-based TCM systems. Therefore, it would be important to carry out more tests, building a new DOE that perhaps includes more levels of machining parameters, allowing a larger orthogonal design. As mentioned earlier, considering the time and material required for the tests carried out in this research, a full factorial design is believed to be too time-consuming. However, it would be desirable to have some repeated tests with the same parameters. This is something that has been mentioned earlier, and it would have helped greatly in understanding the complexity of the ceramic tool's wear, aiding in finding out how unpredictable their wear can really be.

Furthermore, testing with other materials and different machining paths would be important for further research. In the present study, it was shown that in dry machining operations (no coolant), the cutting sparks can give a good relationship with tool wear. In this research, only Waspalloy and Inconel 718 materials were used, and the former was only used in the initial experiment in the author's previous degree. Therefore, this material could be an example of a material for future research, as it was proven in the initial experiments that cutting sparks were very visible and later in section 6.2, that these can be related to tool wear. Therefore, if there are more materials and cutting tools that are commonly used with dry machining, and where cutting sparks are visible, they would definitively be worth exploring. In the case of the cutting tools, there are other ceramic materials that could be interesting for further research, and that perhaps have less complex wear mechanisms. Also, it would have been desirable to carry out tests with all the inserts mounted in the cutting tool. A number of these tests could be very interesting to continue to test the capabilities of the neural networks. Therefore, a full study of how the use of all tools change the results shown in this research would be quite relevant, and perhaps compare them with other tools that can hold a larger number of inserts.

Regarding the flank and notch wear measurement used in section 4.1.1, it would be beneficial to find a more automated way to perform this task. It was mentioned in that section that this task ended up taking considerably more time than was expected, given the number of inserts to measure, and the fact that it was done manually. Therefore, considering the many image processing capabilities of MATLAB and other pieces of software, it should certainly be possible to build a better and faster way of doing this.

Additionally, in that same chapter, the stereo vision approach was described, and the unsuccessful results explained. However, as mentioned previously, this approach has very good potential, if more factors can be added; as the concept of having a system that could give an accurate in-situ direct wear measurement would be quite beneficial. At present, the capabilities of computer vision algorithms for image descriptor recognition, largely used in face recognition systems, could be coupled with a parallel technique to map surface topography. There are systems in the market such as the Microsoft's Kinect, which uses a projected array of infrared points and an infrared camera to compute the depth of a scene. Similarly, other 3D scanners on the market use laser beams to do this scene depth mapping. These scanners are quite expensive devices, but the principles they use are well known. Therefore, with the continuous improvement of image processing techniques and image acquisition capabilities, it is believed that the design of a system like this for tool wear measurement could be a strong possibility.

The objective of chapter 5 was to present a final analysis and selection of the optimal image acquisition systems and parameters for this research. During that analysis, it was considered that cameras can vary in the type of acquisition sensor that they possess, e.g. CCD or CMOS; and that the mechanism of image acquisition, power management, analogue to digital conversion, and other processes can vary. However, even between cameras with similar hardware, the performance of the internal image processors can change. For instance, between SLR cameras, such as the one used in this research, the performance of the ISO parameter, white balance, noise reduction, and so on can different between brands. Therefore, further work could be beneficial to understand the variability that another acquisition system could generate in the results presented.

The absence of literature on spark formation mentioned in section 2.5 also suggested future work in that area. A characterisation and analysis of the sparks generated in the described machining operations would be key to support the hypothesis of this research. The experience extracted from the present work suggests that the spark formation phenomenon can be quite stochastic, which could impact in the repeatability of the testing carried out. Therefore, exploring spark formation could greatly impact in the robustness of the parameters used in this work, as well as any other parameters used in future testing

The general relationship between spark evolution and tool wear was proved in this research. Therefore, a deeper analysis of this relationship would be important, and where the “flaking event” phenomenon could be further explored. It would be interesting to find other ways to correlate this flaking event behaviour in the spark with tool wear. In the present research, the “spark event” input feature was added to the neural networks to improve their accuracy and solve some of the NN limitations. Yet, this spark event identifier could be key to further research on these TCM systems. Perhaps a simpler approach, where a number of these events is taken as a threshold of tool wear could be implemented. Alternatively, perhaps there is a way to quantify these flaking events to understand their magnitude and judge the tool condition through this. Still, this feature was beneficial for the NN performance, and therefore, more research would be required in more tests.

There is definitively much further research regarding the prediction of tool wear with the NN approach. There were some parameters and factors mentioned in section 7.3 that were fixed for all the tests, such as the hyperparameter of weight-decay and the number of training cycles. In the present research, the timeframe limited the possibility of exploring them, and hence, it would be worth exploring their impact on the results, possibly carrying out a cross-validation to select them. Furthermore, as was mentioned towards the end of that chapter, the lower accuracy of the final approach was related to the low number of input examples, which led to an extrapolation task for the NN. Therefore, it would be very important to carry out further testing, so that a wider range of machining conditions can be included during the network's training.

Finally, it would be worth exploring other methods or perhaps combinations of methods of pattern recognition. Even though the NN approach was selected given their high levels of effectiveness, perhaps other methods could treat the confusing effect of the input's outline in a different way. Still, the mentioned further work with NN alone could be quite extended, possibly arriving at an accurate and robust NN, capable of TCM through prediction of tool wear.

8.4 Further Implementation Implications

In the case that the proposed system of TCM was implemented in a realistic scenario, some further implications would be relevant to consider, given the experience and conclusions found by the author throughout this research.

One of these implications would be the situation regarding general lighting conditions. It was described in this research that the initial experiments had somewhat constant lighting conditions, but that the last experiments suffered from a workshop relocation that changed this. Even though there was a methodology revision in section 7.1 to overcome this variation, the general variance of spark feature outline in all tests showed a small noisy fluctuation all the time, regardless of the disengagements and flaking events. This noise may be caused by differences in lighting conditions, and in an industrial scenario, it would be quite expected that lighting conditions may be quite variable. Therefore, this general issue should be further explored to arrive, if necessary, at a more robust method, where this variable has a lower impact.

Furthermore, if a TCM system was implemented as a support system of CNC machine, it would require interaction from the user, so that some inputs can be introduced. In the case of the NN described in chapter 7, these inputs would be the machining parameters. As mentioned previously, the initial approach of individual tests' networks was quite successful, therefore, perhaps in this user interaction, the selection of the required NN could also be a factor.

Also, considering the different algorithm implementations carried out in this research for image processing, it would be expected that a fully functional TCM would require some initial time to auto-adjust. The different colour enhancement and binarization steps described in this work would probably require being adaptive to the lighting conditions, camera orientation and focal parameters. Therefore, it would be important to assure that the camera has a good and constant view of the spark and that the lighting conditions are considered by the system.

Finally, and as mentioned in the initial section of this chapter, such a system would still require a constant human interaction, as the complexity of the ceramics' wear is still an important factor. Furthermore, the working conditions of such a system could include plenty of signal noise, possible machine vibrations and even high or low-temperature conditions. These factors could be important in the general accuracy and robustness of this system, and the human criteria would be beneficial as a parallel condition monitor. Moreover, in the author's opinion, it is believed that an online approach, where the TCM system could communicate directly with the machine, appears not to be achievable with the current results. However, with further work and testing, this may become a possibility. In such a case, the TCM system could communicate with the CNC sequence and automatically decide tool changes and perhaps, even tool paths to improve tool life.

Therefore, this vision-based TCM has plenty of future work, that combined with the findings presented in the present research, could become an essential system for the machining industry.

9 REFERENCES

Addhoum, H. & Broussaud, D., 1989. Interaction of ceramic cutting tools with nickel-based alloys. *Materials Science and Engineering: A*, 109, pp.379–387. Available at: http://www.sciencedirect.com/science/article/pii/0921509389906187 [Accessed November 19, 2015].

Alegre, E. et al., 2009. Use of contour signatures and classification methods to optimize the tool life in metal machining. *Estonian Journal of Engineering*, 15(1), pp.3–12. Available at: http://www.kirj.ee/?id=14991&tpl=1061&c_tpl=1064 [Accessed February 28, 2017].

Altin, A., Nalbant, M. & Taskesen, A., 2007. The effects of cutting speed on tool wear and tool life when machining Inconel 718 with ceramic tools. *Materials & Design*, 28(9), pp.2518–2522. Available at: http://www.sciencedirect.com/science/article/pii/S0261306906002561 [Accessed May 6, 2015].

Altintas, Y., 2000. *Manufacturing Automation: Metal Cutting Mechanics, Machine Tool Vibrations, and CNC Design*, Cambridge University Press. Available at: https://books.google.co.uk/books/about/Manufacturing_Automation.html?id=Xn9Y71zm6JAC&pgis=1 [Accessed May 13, 2015].

Ambhore, N. et al., 2015. Tool condition monitoring system: A review. In *Materials Today: Proceedings*. Elsevier, pp. 3419–3428. Available at: http://linkinghub.elsevier.com/retrieve/pii/S2214785315005623 [Accessed August 15, 2016].

Antony, J., 2003. *Design of Experiments for Engineers and Scientists*, Butterworth-Heinemann. Available at: https://books.google.co.uk/books/about/Design_of_Experiments_for_Engineers_and.html?id=Lg2iyehYXroC&pgis=1 [Accessed October 13, 2015].

Arunachalam, R. & Mannan, M.A., 2000. MACHINABILITY OF NICKEL-BASED HIGH TEMPERATURE ALLOYS. *Machining Science and Technology*, 4(1), pp.127–168.

Astakhov, V.P., 2006. *Tribology of metal cutting*, Elsevier.

Aucote, J. & Foster, S.R., 1986. Performance of sialon cutting tools when machining nickel-base aerospace alloys. *Materials Science and Technology*, 2(7), pp.700–708. Available at: http://www.tandfonline.com/doi/full/10.1179/mst.1986.2.7.700 [Accessed October 28, 2016].

Azmi, A.I., 2015. Monitoring of tool wear using measured machining forces and neuro-fuzzy modelling approaches during machining of GFRP composites. *Advances in Engineering Software*, 82, pp.53–64. Available at: http://www.sciencedirect.com/science/article/pii/S0965997814002105 [Accessed March 2, 2016].

Barreiro, J. et al., 2008. Use of descriptors based on moments from digital images for tool wear monitoring. *International Journal of Machine Tools and Manufacture*, 48(9), pp.1005–1013.

Bhat, N.N. et al., 2016. Tool condition classification in turning process using hidden Markov model based on texture analysis of machined surface images. *Measurement*, 90, pp.500–509. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0263224116301683 [Accessed August 15, 2016].

Bhattacharyya, S.K. et al., 1983. Wear mechanisms of Syalon ceramic tools when machining nickel-based materials. *Metals Technology*, 10(1), pp.482–489. Available at: http://www.tandfonline.com/doi/full/10.1179/030716983803291415 [Accessed February 7, 2017].

Bishop, C.M., 2006. *Pattern recognition and machine learning*, Springer.

Bitterlich, B., Bitsch, S. & Friederich, K., 2008. SiAlON based ceramic cutting tools. *Journal of the European Ceramic Society*, 28(5), pp.989–994. Available at: http://www.sciencedirect.com/science/article/pii/S0955221907004670 [Accessed May 12, 2015].

Bradley, C. & Wong, Y.S., 2001. Surface Texture Indicators of Tool Wear - A Machine Vision Approach. *The International Journal of Advanced Manufacturing Technology*, 17(6), pp.435–443. Available at: http://link.springer.com/10.1007/s001700170161 [Accessed March 1, 2017].

Byrne, G. et al., 1995. Tool Condition Monitoring (TCM) — The Status of Research and Industrial Application. *CIRP Annals - Manufacturing Technology*, 44(2), pp.541–567. Available at: http://www.sciencedirect.com/science/article/pii/S0007850607605034 [Accessed March 21, 2015].

Castejón, M. et al., 2007. On-line tool wear monitoring using geometric descriptors from digital images. *International Journal of Machine Tools and Manufacture*, 47(12–13), pp.1847–1853.

Casto, S.L. et al., 1993. Wear mechanism of ceramic tools. *Wear*, 160(2), pp.227–235. Available at: http://www.sciencedirect.com/science/article/pii/004316489390425L [Accessed May 12, 2015].

Cuevas Jimenez, E.V., Zaldivar Navarro, D. & Perez Cisneros, M.A., 2010. *Procesamiento digital de imágenes con MATLAB y simulación*, RA-MA S.A. Editorial y Publicaciones. Available at: https://books.google.co.uk/books/about/Procesamiento_digital_de_imágenes_con_M.html?id=uVnBcQAACAAJ&pgis=1 [Accessed May 13, 2015].

D'Addona, D.M. & Teti, R., 2013. Image Data Processing via Neural Networks for Tool Wear Prediction. *Procedia CIRP*, 12, pp.252–257. Available at: http://www.sciencedirect.com/science/article/pii/S2212827113006859 [Accessed May 12, 2015].

Dashchenko, A., 2012. *Manufacturing Technologies for Machines of the Future: 21st Century Technologies*, Springer Science & Business Media. Available at: https://books.google.com/books?id=iaDrCAAAQBAJ&pgis=1 [Accessed May 12, 2015].

Dawson, T.G. & Kurfess, T.R., 2005. Quantification of tool wear using white light interferometry and three-dimensional computational metrology. *International Journal of Machine Tools and Manufacture*, 45(4), pp.591–596.

Devillez, A., Lesko, S. & Mozer, W., 2004. Cutting tool crater wear measurement with white light interferometry. *Wear*, 256(1), pp.56–65.

Dimla, D.E. & Lister, P.M., 2000. On-line metal cutting tool condition monitoring.: I: force and vibration analyses. *International Journal of Machine Tools and Manufacture*, 40(5), pp.739–768.

Dominguez Caballero, J.A., 2012. *In process wear assessment of ceramic machining inserts*. The University of Sheffield.

Dominguez Caballero, J.A., Manson, G.A. & Marshall, M.B., 2016a. Optimal Image Processing Acquisition Parameters For A Tool Condition Monitoring System Of Ceramic Inserted Tools. In *XIII International Conference in high Speed Machining*.

Dominguez Caballero, J.A., Manson, G.A. & Marshall, M.B., 2016b. Tool Condition Monitoring of Ceramic Inserted Tools in High Speed Machining through Image Processing. In *World Academy of Science, Engineering and Technology, International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*. pp. 1304–1311.

Flores, F., 2015. Private interview with FRISA employee.

Ghani, J.., Choudhury, I.. & Hassan, H.., 2004. Application of Taguchi method in the optimization of end milling parameters. *Journal of Materials Processing Technology*, 145(1), pp.84–92.

Ghosh, N. et al., 2007. Estimation of tool wear during CNC milling using neural network-based sensor fusion. *Mechanical Systems and Signal Processing*, 21(1), pp.466–479.

Gonzalez, R.C. & Woods, R.E., 2010. *Digital Image Processing: International Edition* 3rd ed. -, ed., Pearson. Available at: http://www.amazon.co.uk/Digital-Image-Processing-International-Edition/dp/0132345633 [Accessed May 12, 2015].

Grzesik, W., 2008. *Advanced Machining Processes of Metallic Materials: Theory, Modelling and Applications* 1st ed., Elsevier. Available at: https://books.google.com/books?id=j-_QA3u1D5EC&pgis=1 [Accessed May 12, 2015].

Haber, R.E. et al., 2004. An investigation of tool-wear monitoring in a high-speed machining process. *Sensors and Actuators A: Physical*, 116(3), pp.539–545.

Kacker, R.N., Lagergren, E.S. & Filliben, J.J., 1991. Taguchi's orthogonal arrays are classical designs of experiments. *Journal of Research of the National Institute of Standards and Technology*, 96(5), pp.577–591. Available at: http://www.ncbi.nlm.nih.gov/pubmed/28184132 [Accessed March 30, 2017].

Kang, M.C., Kim, J.S. & Kim, K.H., 2005. Fractal dimension analysis of machined surface depending on coated tool wear. *Surface and Coatings Technology*, 193(1), pp.259–265.

Kannatey-Asibu, E. & Dornfeld, D.A., 1982. A study of tool wear using statistical analysis of metal-cutting acoustic emission. *Wear*, 76(2), pp.247–261. Available at: http://linkinghub.elsevier.com/retrieve/pii/0043164882900096 [Accessed September 26, 2016].

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

Kassim, A.A., Mian, Z. & Mannan, M.A., 2004. Connectivity oriented fast Hough transform for tool wear monitoring. *Pattern Recognition*, 37(9), pp.1925–1933.

Kassim, A.A., Mian, Z. & Mannan, M.A., 2002. Texture analysis using fractals for tool wear monitoring. In *IEEE International Conference on Image Processing*. IEEE, p. III/105-III/108. Available at: http://ieeexplore.ieee.org/document/1038915/ [Accessed March 1, 2017].

Kassim, A.A., Mian, Z. & Mannan, M.A., 2006. Tool condition classification using Hidden Markov Model based on fractal analysis of machined surface textures. *Machine Vision and Applications*, 17(5), pp.327–336. Available at: http://link.springer.com/10.1007/s00138-006-0038-y [Accessed March 1, 2017].

Ko, T. & Koren, Y., 1989. Cutting force model for tool wear estimation. *Ann Arbor*. Available at: http://trafficlight.bitdefender.com/info?url=http%3A//www-personal.umich.edu/~ykoren/uploads/Cutting_Force_Model_for_Tool_Wear_Estimation.pdf&language=en_US [Accessed May 12, 2015].

Krishnakumar, P., Rameshkumar, K. & Ramachandran, K.I., 2015. Tool Wear Condition Prediction Using Vibration Signals in High Speed Machining (HSM) of Titanium (Ti-6Al-4V) Alloy. *Procedia Computer Science*, 50, pp.270–275. Available at: http://www.sciencedirect.com/science/article/pii/S1877050915005505 [Accessed February 17, 2016].

Kurada, S. & Bradley, C., 1997a. A machine vision system for tool wear assessment. *Tribology International*, 30(4), pp.295–304. Available at: http://www.sciencedirect.com/science/article/pii/S0301679X96000588 [Accessed May 12, 2015].

Kurada, S. & Bradley, C., 1997b. A review of machine vision sensors for tool condition monitoring. *Computers in Industry*, 34(1), pp.55–72. Available at: http://www.sciencedirect.com/science/article/pii/S0166361596000759 [Accessed May 12, 2015].

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

Lee, W.K., Ratnam, M.M. & Ahmad, Z.A., 2016. In-process detection of chipping in ceramic cutting tools during turning of difficult-to-cut material using vision-based approach. *The International Journal of Advanced Manufacturing Technology*, 85(5–8), pp.1275–1290. Available at: http://link.springer.com/10.1007/s00170-015-8038-6 [Accessed August 15, 2016].

Li, W., Singh, H.M. & Guo, Y.B., 2013. An online optical system for inspecting tool condition in milling of H13 tool steel and in 718 alloy. *International Journal of Advanced Manufacturing Technology*, 67(5–8), pp.1067–1077. Available at: http://link.springer.com/10.1007/s00170-012-4548-7 [Accessed September 26, 2016].

Liu, Z.. et al., 2002. Wear patterns and mechanisms of cutting tools in high-speed face milling. *Journal of Materials Processing Technology*, 129(1–3), pp.222–226. Available at: http://www.sciencedirect.com/science/article/pii/S0924013602006052 [Accessed May 12, 2015].

Livingstone, D.J., 2009. *Artificial Neural Networks* D. J. Livingstone, ed., Totowa, NJ: Humana Press.

Mandal, N. et al., 2011. Optimization of flank wear using Zirconia Toughened Alumina (ZTA) cutting tool: Taguchi method and Regression analysis. *Measurement*, 44(10), pp.2149–2155.

Marinescu, I. & Axinte, D., 2009. A time–frequency acoustic emission-based monitoring technique to identify workpiece surface malfunctions in milling with multiple teeth cutting simultaneously. *International Journal of Machine Tools and Manufacture*, 49(1), pp.53–65.

Marinescu, I. & Axinte, D.A., 2008. A critical analysis of effectiveness of acoustic emission signals to detect tool and workpiece malfunctions in milling operations. *International Journal of Machine Tools and Manufacture*, 48(10), pp.1148–1160.

Möhring, H.-C. et al., 2016. Intelligent Tools for Predictive Process Control. *Procedia CIRP*, 57, pp.539–544.

Nabney, I., 2002. *NETLAB : algorithms for pattern recognitions*, Springer.

Nakamura, J., 2005. *Image Sensors and Signal Processing for Digital Still Cameras*, CRC Press. Available at: https://books.google.co.uk/books/about/Image_Sensors_and_Signal_Processing_for.html?id=UY6QzgzgieYC&pgis=1 [Accessed January 25, 2016].

Olufayo, O. & Abou-El-Hossein, K., 2015. Tool life estimation based on acoustic emission monitoring in end-milling of H13 mould-steel. *The International Journal of Advanced Manufacturing Technology*, 81(1–4), pp.39–51. Available at: http://link.springer.com/10.1007/s00170-015-7091-5 [Accessed February 24, 2017].

Orhan, S. et al., 2007. Tool wear evaluation by vibration analysis during end milling of AISI D3 cold work tool steel with 35 HRC hardness. *NDT & E International*, 40(2), pp.121–126.

Otieno, A. et al., 2006. Imaging and wear analysis of micro-tools using machine vision. In *Proceedings of the IJME*. Available at: http://ijme.us/cd_06/PDF/IT 301-071.pdf [Accessed February 28, 2017].

Pawade, R.S. & Joshi, S.S., 2012. Analysis of acoustic emission signals and surface integrity in the high-speed turning of Inconel 718. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 226(1), pp.3–27. Available at: http://pib.sagepub.com/lookup/doi/10.1177/0954405411407656 [Accessed February 24, 2017].

Pfeifer, T. & Wiegers, L., 2000. Reliable tool wear monitoring by optimized image and illumination control in machine vision. *Measurement*, 28(3), pp.209–218. Available at: http://www.sciencedirect.com/science/article/pii/S0263224100000142 [Accessed May 12, 2015].

Ren, Q. et al., 2014. Type-2 fuzzy tool condition monitoring system based on acoustic emission in micromilling. *Information Sciences*, 255, pp.121–134.

Renz, A., Khader, I. & Kailer, A., 2015. Tribochemical wear of cutting-tool ceramics in sliding contact against a nickel-base alloy. *Journal of the European Ceramic Society*, 36(3), pp.705–717. Available at: http://www.sciencedirect.com/science/article/pii/S0955221915301953 [Accessed November 19, 2015].

Richards, N. & Aspinwall, D., 1989. Use of ceramic tools for machining nickel based alloys. *International Journal of Machine Tools and Manufacture*, 29(4), pp.575–588. Available at: http://www.sciencedirect.com/science/article/pii/0890695589900722.

Sandvik Coromant, 2010. *Ceramics*, Available at: http://www.sandvik.coromant.com/sitecollectiondocuments/downloads/global/brochures/en-gb/c-2929-61.pdf [Accessed February 3, 2014].

Sandvik Coromant, *Heat resistant super alloys – HRSA*, Available at: http://www.sandvik.coromant.com/SiteCollectionDocuments/downloads/global/technical guides/en-us/C-2920-034.pdf [Accessed March 14, 2014].

Sandvik Coromant, 1994. *Modern metal cutting: a practical handbook* 1st ed., Sandvik Coromant. Available at: https://books.google.co.uk/books/about/Modern_metal_cutting.html?id=aOlSAAAAMAAJ&pgis=1 [Accessed May 13, 2015].

Sevilla-Camacho, P.Y. et al., 2015. FPGA-based reconfigurable system for tool condition monitoring in high-speed machining process. *Measurement*, 64, pp.81–88. Available at: http://www.sciencedirect.com/science/article/pii/S0263224114006368 [Accessed February 26, 2016].

Szeliski, R., 2010. *Computer Vision: Algorithms and Applications*, Springer Science & Business Media. Available at: https://books.google.com/books?hl=en&lr=&id=bXzAlkODwa8C&pgis=1 [Accessed May 12, 2015].

Teshima, T. et al., 1993. Estimation of Cutting Tool Life by Processing Tool Image Data with Neural Network. *CIRP Annals - Manufacturing Technology*, 42(1), pp.59–62. Available at: http://www.sciencedirect.com/science/article/pii/S0007850607623919 [Accessed May 12, 2015].

Tian, X. et al., 2013. A comparison between whisker-reinforced alumina and SiAlON ceramic tools in high-speed face milling of Inconel 718. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 228(8), pp.845–857. Available at: http://pib.sagepub.com/content/228/8/845 [Accessed November 30, 2015].

Tian, X. et al., 2013. Effect of cutting speed on cutting forces and wear mechanisms in high-speed face milling of Inconel 718 with Sialon ceramic tools. *The International Journal of Advanced Manufacturing Technology*, 69(9–12), pp.2669–2678. Available at: http://link.springer.com/10.1007/s00170-013-5206-4 [Accessed November 30, 2015].

Venkata Rao, K., Murthy, B.S.N. & Mohan Rao, N., 2013. Cutting tool condition monitoring by analyzing surface roughness, work piece vibration and volume of metal removed for AISI 1040 steel in boring. *Measurement*, 46(10), pp.4075–4084.

Wang, G. et al., 2014. Force based tool wear monitoring system for milling process based on relevance vector machine. *Advances in Engineering Software*, 71, pp.46–51.

Wang, G.F. et al., 2014. Vibration sensor based tool condition monitoring using v support vector machine and locality preserving projection. *Sensors and Actuators, A: Physical*, 209, pp.24–32.

Wang, W., Wong, Y.S. & Hong, G.S., 2005. Flank wear measurement by successive image analysis. *Computers in Industry*, 56(8), pp.816–830.

Wang, W.H., Hong, G.S. & Wong, Y.S., 2006. Flank wear measurement by a threshold independent method with sub-pixel accuracy. *International Journal of Machine Tools and Manufacture*, 46(2), pp.199–207.

Wang, W.H., Wong, Y.S. & Hong, G.S., 2006. 3D measurement of crater wear by phase shifting method. *Wear*, 261(2), pp.164–171.

Weis, W., 1993. Tool wear measurement on basis of optical sensors, vision systems and neuronal networks (application milling). *Proceedings of WESCON '93*, pp.134–138. Available at: http://ieeexplore.ieee.org/document/488423/ [Accessed February 28, 2017].

Xiong, G., Liu, J. & Avila, A., 2011. Cutting tool wear measurement by using active contour model based image processing. *2011 IEEE International Conference on Mechatronics and Automation*, 2, pp.670–675. Available at: http://ieeexplore.ieee.org/document/5985741/ [Accessed February 28, 2017].

Yadav, N., Yadav, A. & Kumar, M., 2015. *An Introduction to Neural Network Methods for Differential Equations*, Dordrecht: Springer Netherlands.

Zhang, C. & Zhang, J., 2013. On-line tool wear measurement for ball-end milling cutter based on machine vision. *Computers in Industry*, 64(6), pp.708–719. Available at: http://www.sciencedirect.com/science/article/pii/S0166361513000638 [Accessed April 9, 2015].

Zhang, S., Li, J.F. & Wang, Y.W., 2012. Tool life and cutting forces in end milling Inconel 718 under dry and minimum quantity cooling lubrication cutting conditions. *Journal of Cleaner Production*, 32, pp.81–87.

Zheng, G. et al., 2016. Experimental Investigation on Sialon Ceramic Inserts for Ultra-high-speed Milling of Inconel 718. *Materials and Manufacturing Processes*, 6914(July), pp.633–640. Available at: http://www.tandfonline.com/doi/full/10.1080/10426914.2015.1019090 [Accessed October 18, 2016].

Zheng, G., Zhao, J. & Zhou, Y., 2012. Friction and wear behaviors of Sialon–Si3N4 graded nano-composite ceramic materials in sliding wear tests and in cutting processes. *Wear*, 290–291, pp.41–50. Available at: http://www.sciencedirect.com/science/article/pii/S0043164812001950 [Accessed May 12, 2015].

Zhou, J.-H. et al., 2011. Tool Wear Monitoring Using Acoustic Emissions by Dominant-Feature Identification. *IEEE Transactions on Instrumentation and Measurement*, 60(2), pp.547–559. Available at: http://ieeexplore.ieee.org/document/5483234/ [Accessed February 24, 2017].

Zhu, D., Zhang, X. & Ding, H., 2013. Tool wear characteristics in machining of nickel-based superalloys. *International Journal of Machine Tools and Manufacture*, 64, pp.60–77. Available at: http://www.sciencedirect.com/science/article/pii/S0890695512001575 [Accessed June 24, 2015].

Zhu, K. & Yu, X., 2017. The monitoring of micro milling tool wear conditions by wear area estimation. *Mechanical Systems and Signal Processing*, 93, pp.80–91.

APPENDIX I – CNC PROGRAMMES FOR EXPERIMENTS

CNC programme for machining of square shape for tests:

```
G17
 G90
 G71
 T="CERAMIC INSERT" M6
 G55 S5000 M3
 G0 X-110.892 Y-105.013
 Z5
 Z-0.5
 G1 G94 Z-1.5 F100
 X-110.167 Y-104.288 F750
 X-110.542 Y-103.752
 X-111.046 Y-102.974
 X-111.632 Y-101.974
 G2 X-113.054 Y-98.975 I21.968 J12.248
 G1 X-113.429 Y-97.975
 X-113.749 Y-96.976
 G2 X-114.452 Y-93.976 I26.743 J7.851
 X-114.803 Y-89.977 I26.955 J4.383
 G1 Y89.977
 G2 X-114.6 Y92.977 I29.781 J-0.51
 G1 X-114.452 Y93.976
 G2 X-113.749 Y96.976 I27.446 J-4.851
 G1 X-113.429 Y97.975
 G2 X-112.64 Y99.975 I29.96 J-10.666
 G1 X-112.163 Y100.975
 X-111.972 Y101.354
 X-111.632 Y101.974
 X-111.046 Y102.974
 X-110.607 Y103.654
 X-110.18 Y104.274
 X-110.905 Y104.999
 G0 Z5
 X-110.889 Y-105.028
 Z-2
 G1 Z-3 F100
 X-110.163 Y-104.303 F750
 G2 X-111.05 Y-102.974 I22.025 J15.654
```

CNC programme for tests:

```
G17
 G90
 G71
 T="CERAMIC INSERT" M6
 G55 S5411 M3
 G0 X-74.7 Y125
 Z0
```

```
BEGIN: G1 Y-125 F595
G0 Z50 M0
Y125 M3
G91 X40.05
END: G90 Z0
REPEAT BEGIN END P=3
G0 Z200
TRAFOOF
ROT
TRANS
M30
```

APPENDIX II – CRATER WEAR AREA ALGORITHM

```
18/09/17 12:49   D:\MATLAB\Main Te...\Insert Crater Wear.m   1 of 3

%%%%%%%%%%%%%%%%%%%%Inserts Crater Wear Area calcualtion.%%%%%%%%%%%%%%%%%%%%
clearvars, clc, close all,
prompt = 'Number of insert images to process?   ';
NumFotos = input(prompt);
MaxAreamm = cast([], 'double');
for i = 1:NumFotos
    close all;
    prompt = 'What is the number in the inserts file name? (e.g. for Image205.jpg, write = 205)   ';
    NumInserts = input(prompt);
    filename = sprintf('Image%01d.jpg', NumInserts);
    Insert = imread(filename);
    Icrop = imcrop(Insert);
    figure;
    Iextract = imcrop(Icrop);
    R = Iextract(:,:,1);
    G = Iextract(:,:,2);
    B = Iextract(:,:,3);

    Rmaxmean = mean(max(R));
    Gmaxmean = mean(max(G));
    Bmaxmean = mean(max(B));

    Rminmean = mean(min(R));
    Gminmean = mean(min(G));
    Bminmean = mean(min(B));

    Rlowin = Rminmean/255;
    Glowin = Gminmean/255;
    Blowin = Bminmean/255;
    Rhighin = Rmaxmean/255;
    Ghighin = Gmaxmean/255;
    Bhighin = Bmaxmean/255;

    AdjI = imadjust(Icrop,[Rlowin Glowin Blowin; Rhighin Ghighin Bhighin],[]);
    figure;
    imshow(AdjI);
    thresh = 0.05;
    step = 0.05;
    figure;
    for k = 1:18
        bwlevel = thresh;
        BwI = im2bw(AdjI,bwlevel);
        subplot(3,6,k);
        imshow(BwI);
        title(thresh);
        thresh = thresh + step;
    end
    response = input('Were the enhancement and thresholds correct? Y/N:  ','s');
    if strcmp(response,'Y')
        prompt = 'What was the best threshold? ';
        bwlevel = input(prompt);
        BwI = im2bw(AdjI,bwlevel);
        BwII = imcomplement(BwI);
        figure;
        imshow(BwII);
        BW2 = imfill(BwII,'holes');
        figure;
        imshow(BW2);
        BWareas = bwareaopen(BW2, 50);
        CC = bwconncomp(BWareas, 4);
        CC.NumObjects;
        ImageData = regionprops(CC, 'basic');
        Areas = [ImageData.Area];
        [MaxArea,idx] = max(Areas);
        Wear = false(size(BW2));
        Wear(CC.PixelIdxList{idx}) = true;
        figure;
        imshow(Wear);
        figure;
        Imeasure = imcrop(Insert);
        close figure 6;
        Widthpix = size(Imeasure,2);
        Widthmm = 12.54/Widthpix;
        Areapix = Widthmm*Widthmm;
        MaxAreamm(i) = MaxArea*Areapix;
    else
        while strcmp(response,'N')
            close figure 2; close figure 3; close figure 4;
            figure;
            Iextract = imcrop(Icrop);
            R = Iextract(:,:,1);
            G = Iextract(:,:,2);
            B = Iextract(:,:,3);

            Rmaxmean = mean(max(R));
            Gmaxmean = mean(max(G));
            Bmaxmean = mean(max(B));

            Rminmean = mean(min(R));
            Gminmean = mean(min(G));
            Bminmean = mean(min(B));

            Rlowin = Rminmean/255;
            Glowin = Gminmean/255;
            Blowin = Bminmean/255;
            Rhighin = Rmaxmean/255;
            Ghighin = Gmaxmean/255;
            Bhighin = Bmaxmean/255;

            AdjI = imadjust(Icrop,[Rlowin Glowin Blowin; Rhighin Ghighin Bhighin],↙
[]);
            figure;
            imshow(AdjI);
            thresh = 0.05;
            step = 0.05;
            figure;
            for k = 1:18
                bwlevel = thresh;
                BwI = im2bw(AdjI,bwlevel);
```

18/09/17 12:49   D:\MATLAB\Main Te...\Insert Crater Wear.m   3 of 3

```
                subplot(3,6,k);
                imshow(BwI);
                title(thresh);
                thresh = thresh + step;
            end
            response = input('Were the enhancement and thresholds correct? Y/N:↙
','s');
        end
        prompt = 'What was the best threshold? ';
        bwlevel = input(prompt);
        BwI = im2bw(AdjI,bwlevel);
        BwII = imcomplement(BwI);
        figure;
        imshow(BwII);
        BW2 = imfill(BwII,'holes');
        figure;
        imshow(BW2);
        BWareas = bwareaopen(BW2, 50);
        CC = bwconncomp(BWareas, 4);
        CC.NumObjects;
        ImageData = regionprops(CC, 'basic');
        Areas = [ImageData.Area];
        MaxArea = max(Areas);
        figure;
        Imeasure = imcrop(Insert);
        close figure 6;
        Widthpix = size(Imeasure,2);
        Widthmm = 12.54/Widthpix;
        Areapix = Widthmm*Widthmm;
        MaxAreamm(i) = MaxArea*Areapix;
    end
end
plot(MaxAreamm);
```

APPENDIX III – STEREO VISION ALGORITHM FOR 3D RECONSTRUCTION.

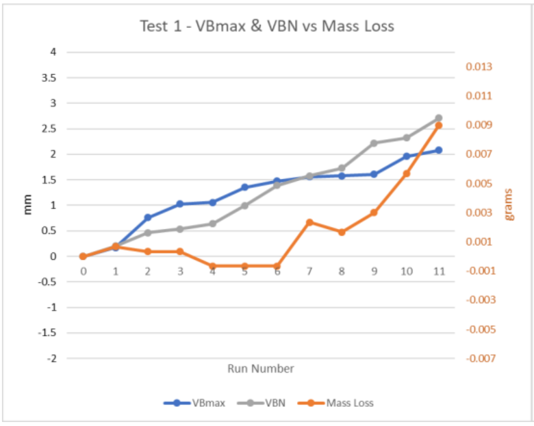
18/09/17 13:03   D:\MATLAB\Stereo...\Stereo Vision Test4.m   1 of 2

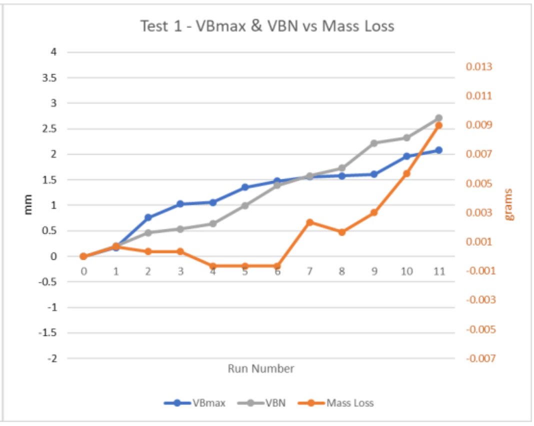
```
 1 clearvars; clc; close all;
 2 
 3 % Full Stereo Vision Algorithm (to support this algorithm findings, the
 4 % MATLAB's "Stereo Calibration App" was also used, replacing lines from
 5 % 7 to 43).
 6 
 7 numImagePairs = 38 ;
 8 allimagesleft = cast([], 'uint8');
 9 allimagesright= cast([], 'uint8');
10 for i = 1:numImagePairs
11     filename = sprintf('IMG_%04d.JPG', i);
12     im = imread(filename);
13     allimagesleft(:, :, :, i) = imadjust(rgb2gray(imcrop(im, [0 0 2592 2912])));
14     filename = sprintf('left%02d.jpg', i);
15     imwrite(allimagesleft(:, :, :, i), filename);
16     allimagesright(:, :, :, i) = imadjust(rgb2gray(imcrop(im, [2593 0 2592↙
2912])));
17     filename = sprintf('right%02d.jpg', i);
18     imwrite(allimagesright(:, :, :, i), filename);
19 end
20 
21 % Try to detect the checkerboard
22 [imagePoints, boardSize, pairsUsed] = detectCheckerboardPoints(allimagesleft,↙
allimagesright);
23 
24 % Display one masked image with the correctly detected checkerboard
25 k = find(pairsUsed);
26 figure;
27 subplot(1,2,1);
28 imshow(allimagesleft(:, :, :,k(1)), 'InitialMagnification', 30);
29 hold on;
30 plot(imagePoints(:, 1, 1, 1), imagePoints(:, 2, 1, 1), '*-g');
31 subplot(1,2,2);
32 imshow(allimagesright(:, :, :,k(1)), 'InitialMagnification', 30);
33 hold on;
34 plot(imagePoints(:, 1, 1, 2), imagePoints(:, 2, 1, 2), '*-g');
35 annotation('textbox', [0 0.9 1 0.1], 'String', 'Successful Checkerboard Detection',↙
'EdgeColor', 'none', 'HorizontalAlignment', 'center');
36 
37 
38 % Generate world coordinates of the checkerboard points.
39 squareSize = 3; % millimeters
40 worldPoints = generateCheckerboardPoints(boardSize, squareSize);
41 
42 % Compute the stereo camera parameters.
43 stereoParams = estimateCameraParameters(imagePoints, worldPoints);
44 
45 % Evaluate calibration accuracy. With the Stereo Calibration App, the
46 % stereoParams and worldPoints were exported and this programme was used
47 % from this point onwards.
48 
49 figure;
50 showReprojectionErrors(stereoParams);
51 
52 I1 = imread('left38rgb.png');
```

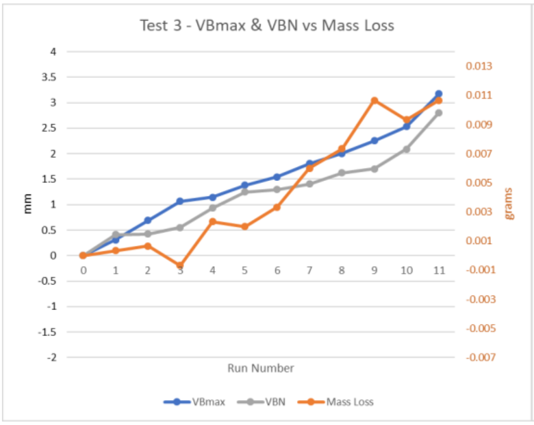
18/09/17 13:03 D:\MATLAB\Stereo...\Stereo Vision Test4.m 2 of 2

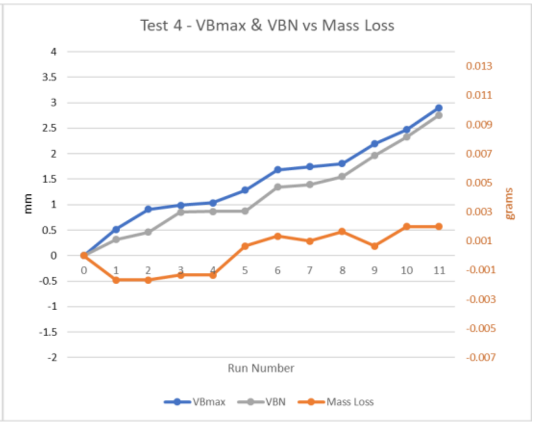
```
I2 = imread('right38rgb.png');

% Rectify the images.
[J1, J2] = rectifyStereoImages(I1, I2, stereoParams, 'OutputView', 'full');

% Display the images before rectification.
figure;
imshow(stereoAnaglyph(I1, I2), 'InitialMagnification', 30);
title('Before Rectification');

% Display the images after rectification.
figure;
imshow(stereoAnaglyph(J1, J2), 'InitialMagnification', 30);
title('After Rectification');

disparityRange = [-32, 928];
disparityMap = disparity(rgb2gray(J1), rgb2gray(J2), 'DistanceThreshold', 2,↙
'UniquenessThreshold', 2, 'BlockSize', 15, 'DisparityRange', ...
    disparityRange);
figure;
imshow(disparityMap, disparityRange, 'InitialMagnification', 30);
colormap('jet');
colorbar;
title('Disparity Map');
point3D = reconstructScene(disparityMap, stereoParams);


z = point3D(:, :, 3);
maxZ = 390;
minZ = 340;
zdisp = z;
zdisp(z < minZ | z > maxZ) = NaN;
point3Ddisp = point3D;
point3Ddisp(:,:,3) = zdisp;
figure
pcshow(point3Ddisp, J1, 'VerticalAxis', 'Y', 'VerticalAxisDir', 'Down' );
xlabel('X');
ylabel('Y');
zlabel('Z');
```

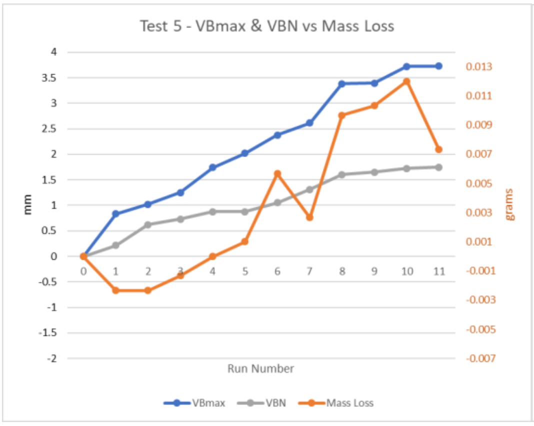
APPENDIX IV – INDIVIDUAL TESTS' TOOL WEAR RESULTS

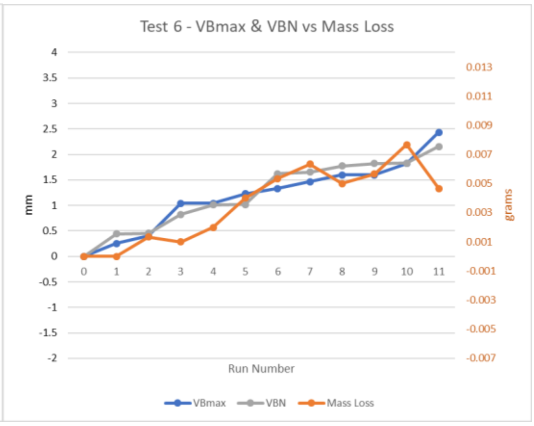




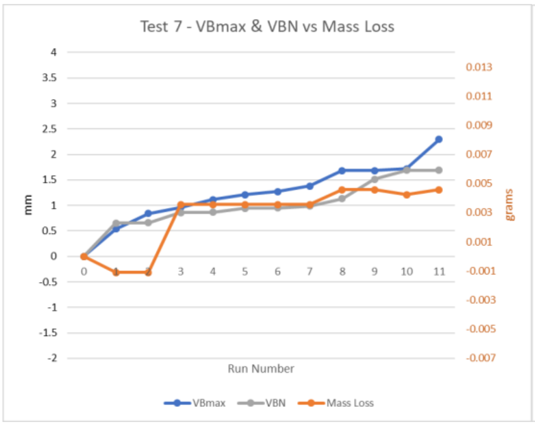


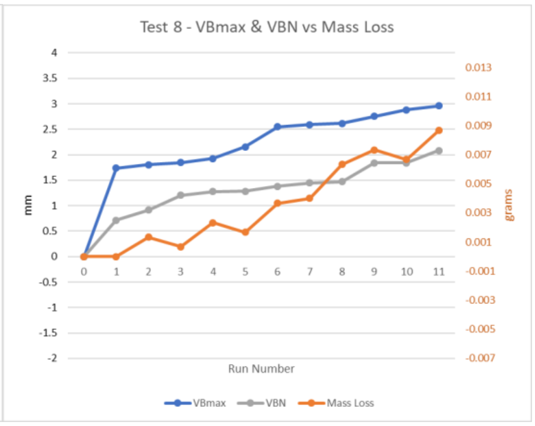


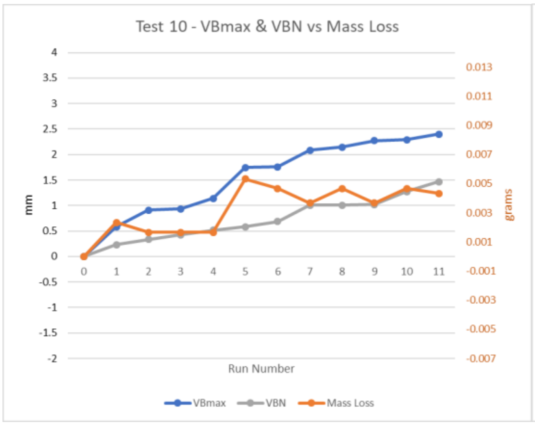


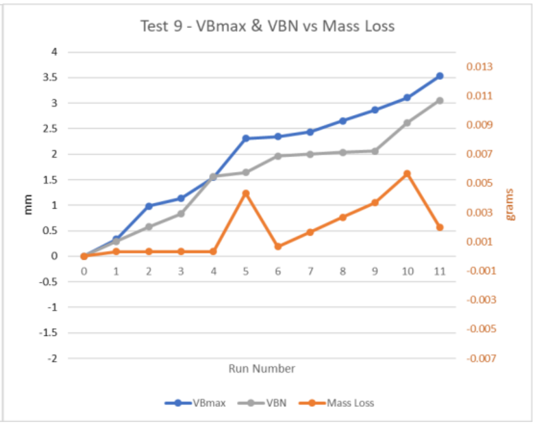


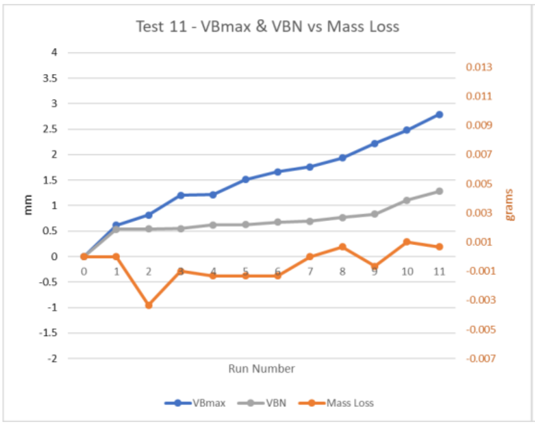
APPENDIX IV – INDIVIDUAL TESTS' TOOL WEAR RESULTS

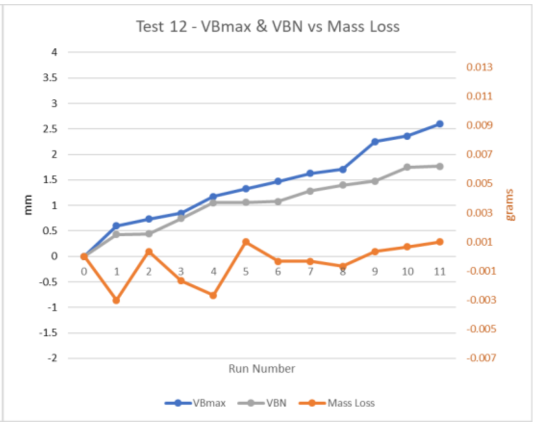


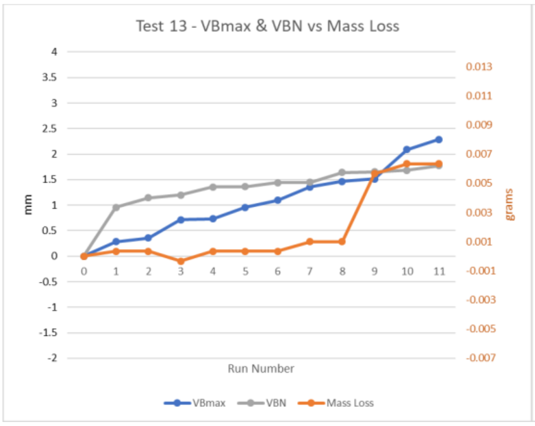


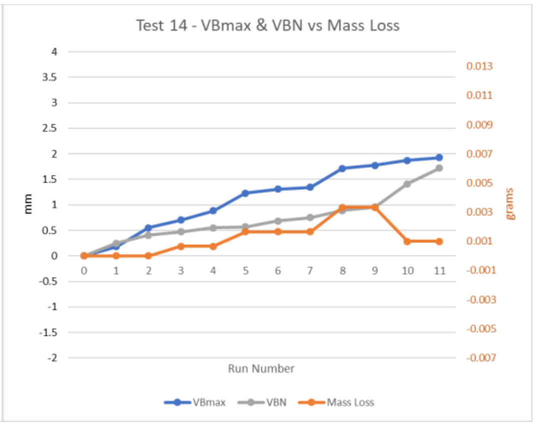


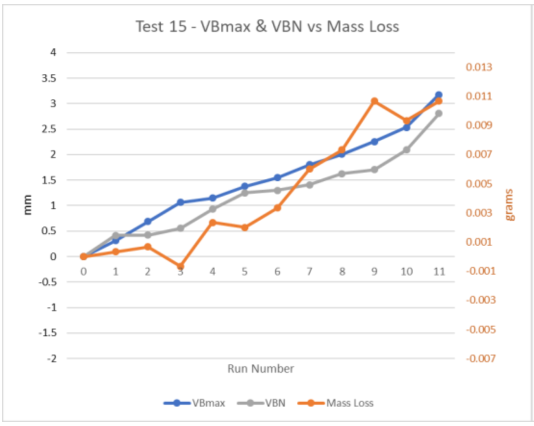


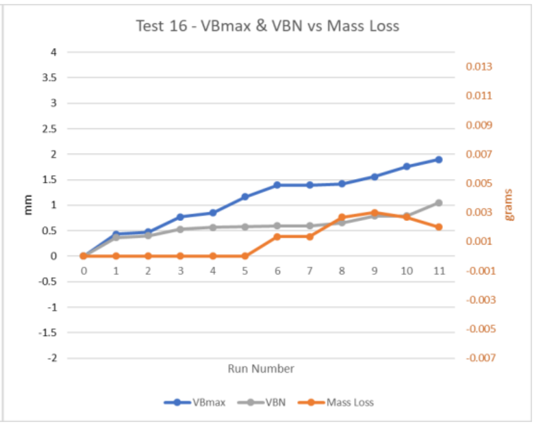


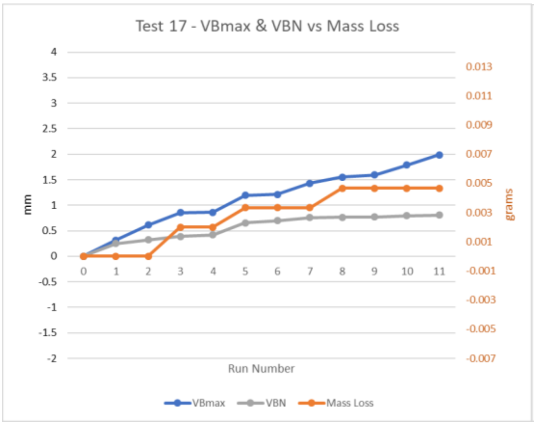


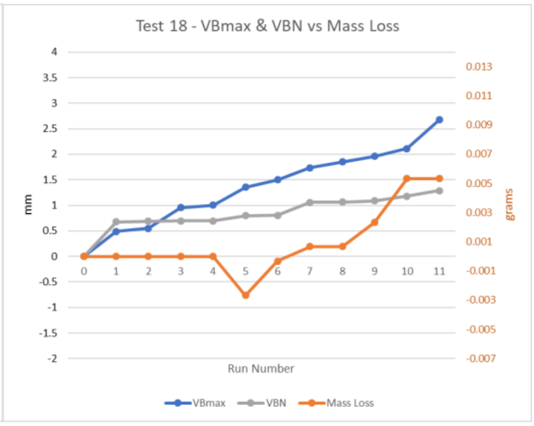




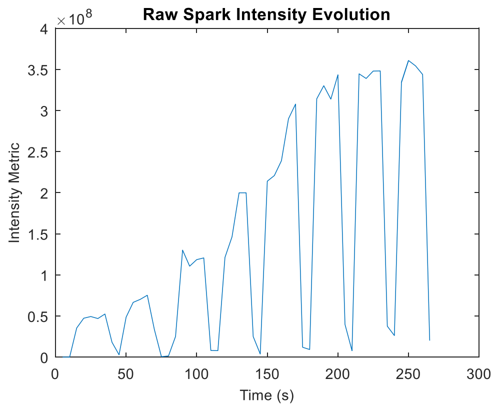

Test 15 - VBmax & VBN vs Mass Loss
mm
grams
Run Number
VBmax
VBN
Mass Loss

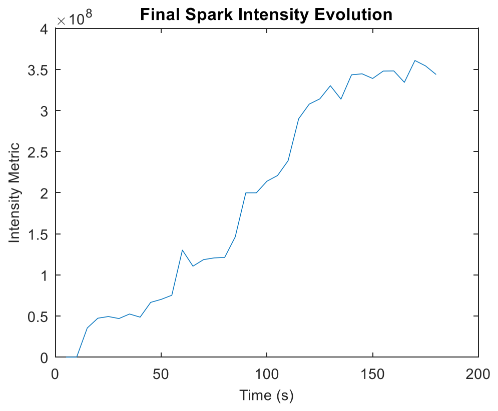

Test 16 - VBmax & VBN vs Mass Loss
mm
grams
Run Number
VBmax
VBN
Mass Loss

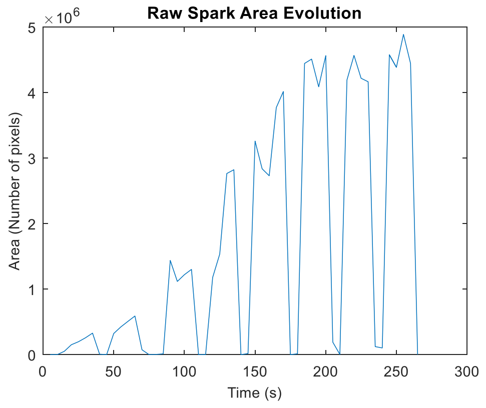

Test 17 - VBmax & VBN vs Mass Loss
mm
grams
Run Number
VBmax
VBN
Mass Loss

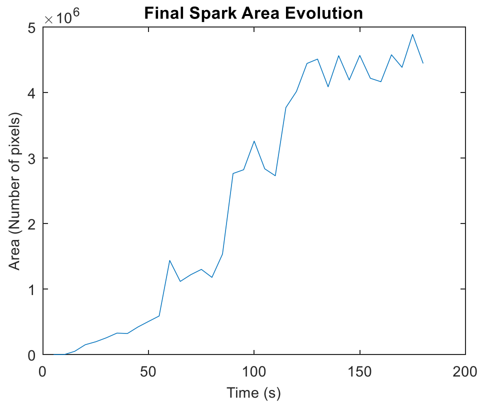

Test 18 - VBmax & VBN vs Mass Loss
mm
grams
Run Number
VBmax
VBN
Mass Loss

APPENDIX V – RAW AND FINAL RESULTS OF FIRST EXPERIMENTS.









APPENDIX VI – ALGORITHM OF FIRST SET OF EXPERIMENTS

```
clear variables; close all; clc;

%%%%%%%%%%%%%%%%%%%%Inserts Crater Wear Area calcualtion.%%%%%%%%%%%%%%%%%%%%
MaxArea(1) = 0;
InsertIndex = 2;
Insert1 = imread ('1T.JPG');
AdjI1 = imadjust(Insert1,[.03 .024 .009; .252 .215 .187],[]);
CropAI1 = AdjI1(310:390,250:530,:);
GrayCAI1 = rgb2gray(CropAI1);
AdjGCAI1 = imadjust(GrayCAI1);
bwlevel = 0.6; %Creating binary image
BWAGCAI1 = im2bw(AdjGCAI1,bwlevel);
InvBAGCAI1 = imcomplement(BWAGCAI1);
FiltIBAGCAI1 = bwareaopen(InvBAGCAI1, 50);
ConCompFIBAGCAI1 = bwconncomp(InvBAGCAI1, 4);
ConCompFIBAGCAI1.NumObjects;
ImageData1 = regionprops(ConCompFIBAGCAI1, 'basic');
Areas1 = [ImageData1.Area];
MaxArea(InsertIndex) = max(Areas1);
% clearvars -except MaxArea Insert1;

InsertIndex = 3;
Insert2 = imread ('2T.JPG');
AdjI2 = imadjust(Insert2,[.2 .16 .121; .323 .29 .267],[]);
CropAI2 = AdjI2(275:390,250:550,:);
GrayCAI2 = rgb2gray(CropAI2);
AdjGCAI2 = imadjust(GrayCAI2);
bwlevel = 0.8; %Creating binary image
BWAGCAI2 = im2bw(AdjGCAI2,bwlevel);
InvBAGCAI2 = imcomplement(BWAGCAI2);
FiltIBAGCAI2 = bwareaopen(InvBAGCAI2, 50);
ConCompFIBAGCAI2 = bwconncomp(InvBAGCAI2, 4);
ConCompFIBAGCAI2.NumObjects;
ImageData2 = regionprops(ConCompFIBAGCAI2, 'basic');
Areas2 = [ImageData2.Area];
MaxArea(InsertIndex) = max(Areas2);
% clearvars -except MaxArea Insert1;

InsertIndex = 4;
Insert3 = imread ('3T.JPG');
AdjI3 = imadjust(Insert3,[.242 .218 .183; .262 .246 .224],[]);
CropAI3 = AdjI3(270:395,235:545,:);
GrayCAI3 = rgb2gray(CropAI3);
AdjGCAI3 = imadjust(GrayCAI3);
bwlevel = 0.9; %Creating binary image
BWAGCAI3 = im2bw(AdjGCAI3,bwlevel);
InvBAGCAI3 = imcomplement(BWAGCAI3);
FiltIBAGCAI3 = bwareaopen(InvBAGCAI3, 50);
ConCompFIBAGCAI3 = bwconncomp(InvBAGCAI3, 4);
ConCompFIBAGCAI3.NumObjects;
ImageData3 = regionprops(ConCompFIBAGCAI3, 'basic');
Areas3 = [ImageData3.Area];
MaxArea(InsertIndex) = max(Areas3);
% clearvars -except MaxArea Insert1;

```

18/09/17 13:26 C...\SparkAreaAndIntensity With Inserts.m   2 of 5

```
InsertIndex = 5;
Insert4 = imread ('4T.JPG');
AdjI4 = imadjust(Insert4,[.36 .32 .3; .448 .419 .392],[]);
CropAI4 = AdjI4(245:390,280:580,:);
GrayCAI4 = rgb2gray(CropAI4);
AdjGCAI4 = imadjust(GrayCAI4);
bwlevel = 0.3; %Creating binary image
BWAGCAI4 = im2bw(AdjGCAI4,bwlevel);
InvBAGCAI4 = imcomplement(BWAGCAI4);
FiltIBAGCAI4 = bwareaopen(InvBAGCAI4, 50);
ConCompFIBAGCAI4 = bwconncomp(InvBAGCAI4, 4);
ConCompFIBAGCAI4.NumObjects;
ImageData4 = regionprops(ConCompFIBAGCAI4, 'basic');
Areas4 = [ImageData4.Area];
MaxArea(InsertIndex) = max(Areas4);
% clearvars -except MaxArea Insert1;

InsertIndex = 6;
Insert5 = imread ('5T.JPG');
AdjI5 = imadjust(Insert5,[.38 .35 .33; .448 .425 .399],[]);
CropAI5 = AdjI5(250:395,220:525,:);
GrayCAI5 = rgb2gray(CropAI5);
AdjGCAI5 = imadjust(GrayCAI5);
bwlevel = 0.3; %Creating binary image
BWAGCAI5 = im2bw(AdjGCAI5,bwlevel);
InvBAGCAI5 = imcomplement(BWAGCAI5);
FiltIBAGCAI5 = bwareaopen(InvBAGCAI5, 50);
ConCompFIBAGCAI5 = bwconncomp(InvBAGCAI5, 4);
ConCompFIBAGCAI5.NumObjects;
ImageData5 = regionprops(ConCompFIBAGCAI5, 'basic');
Areas5 = [ImageData5.Area];
MaxArea(InsertIndex) = max(Areas5);
% clearvars -except MaxArea Insert1;

InsertIndex = 7;
Insert6 = imread ('6T.JPG');
AdjI6 = imadjust(Insert6,[.366 .342 .285; .601 .552 .449],[]);
CropAI6 = AdjI6(210:375,250:600,:);
GrayCAI6 = rgb2gray(CropAI6);
AdjGCAI6 = imadjust(GrayCAI6);
bwlevel = 0.3; %Creating binary image
BWAGCAI6 = im2bw(AdjGCAI6,bwlevel);
InvBAGCAI6 = imcomplement(BWAGCAI6);
FiltIBAGCAI6 = bwareaopen(InvBAGCAI6, 50);
ConCompFIBAGCAI6 = bwconncomp(InvBAGCAI6, 4);
ConCompFIBAGCAI6.NumObjects;
ImageData6 = regionprops(ConCompFIBAGCAI6, 'basic');
Areas6 = [ImageData6.Area];
MaxArea(InsertIndex) = max(Areas6);
% clearvars -except MaxArea Insert1;

InsertIndex = 8;
Insert7 = imread ('7T.JPG');
AdjI7 = imadjust(Insert7,[.212 .174 .144; .49 .452 .426],[]);
CropAI7 = AdjI7(200:400,300:615,:);
```

```
GrayCAI7 = rgb2gray(CropAI7);
AdjGCAI7 = imadjust(GrayCAI7);
bwlevel = 0.8; %Creating binary image
BWAGCAI7 = im2bw(AdjGCAI7,bwlevel);
InvBAGCAI7 = imcomplement(BWAGCAI7);
FiltIBAGCAI7 = bwareaopen(InvBAGCAI7, 50);
ConCompFIBAGCAI7 = bwconncomp(InvBAGCAI7, 4);
ConCompFIBAGCAI7.NumObjects;
ImageData7 = regionprops(ConCompFIBAGCAI7, 'basic');
Areas7 = [ImageData7.Area];
MaxArea(InsertIndex) = max(Areas7);
% clearvars -except MaxArea Insert1;

InsertIndex = 9;
Insert8 = imread ('8T.JPG');
AdjI8 = imadjust(Insert8,[.208 .192 .198; .405 .366 .337],[]);
CropAI8 = AdjI8(170:380,300:610,:);
GrayCAI8 = rgb2gray(CropAI8);
AdjGCAI8 = imadjust(GrayCAI8);
bwlevel = 0.95; %Creating binary image
BWAGCAI8 = im2bw(AdjGCAI8,bwlevel);
InvBAGCAI8 = imcomplement(BWAGCAI8);
FiltIBAGCAI8 = bwareaopen(InvBAGCAI8, 50);
ConCompFIBAGCAI8 = bwconncomp(InvBAGCAI8, 4);
ConCompFIBAGCAI8.NumObjects;
ImageData8 = regionprops(ConCompFIBAGCAI8, 'basic');
Areas8 = [ImageData8.Area];
MaxArea(InsertIndex) = max(Areas8);
% clearvars -except MaxArea InsertIndex Insert1;

%%%%%%%%%%%%%%%%%%%Spark Intensity and Area Extraction.%%%%%%%%%%%%%%%%%%%%%%%%
TotalNumImages = 53;
RefI = imread('IMG_0001.JPG');
m = 2;
n = 2;

%Intensity Variables.
RawIntensity = cast([], 'double');
FinalIntensity = cast([], 'double');
LPF = zeros(3456,5184);
FSize = 80;
LPF(1729-(FSize/2):1729+(FSize/2),2593-(FSize/2):2593+(FSize/2))= 1;
RawIntensity(1) = 0;
FinalIntensity(1) = 0;

%Area Variables
RawArea = cast([], 'double');
FinalArea = cast([], 'double');
AdjRI = imadjust(RefI,[.7 .7 .8; .9 .9 .9],[]);
GrayARI = rgb2gray(AdjRI);
RawArea(1) = 0;
FinalArea(1) = 0;

for k = 2:TotalNumImages
    filename = sprintf('IMG_%04d.JPG', k);
    Image = imread(filename);
    %Intensity processing.
    SubtrSI = imsubtract(Image,RefI);
    AdjSSI = imadjust(SubtrSI,[.3 .4 .6; .9 .9 .8],[]);
    MeanASSI = mean(AdjSSI,3);
    FourierMASSI = fftshift(fft2(MeanASSI));
    FilterFMASSI = FourierMASSI.*LPF;
    InvfourierFFMASSI = ifft2(ifftshift(FilterFMASSI));
    AbsIFFMASSI = abs(InvfourierFFMASSI);
    IntensityAIFFMASSI = sum(sum(AbsIFFMASSI));
    RawIntensity(k) = IntensityAIFFMASSI;

    %Area processing.
    AdjAreaI = imadjust(Image,[.7 .7 .7; .9 .9 .9],[]);
    GrayAAI = rgb2gray(AdjAreaI);
    SubtrGAAI = imsubtract(GrayAAI,GrayARI);
    bwlevel = 0.04;
    BwSGAAI = im2bw(SubtrGAAI,bwlevel);
    Imall(:, :, k-1) = BwSGAAI;
%     thresh = thresh + 0.019;
    FiltBSGAAI = bwareaopen(BwSGAAI, 50);
    ConCompFBSGAAI = bwconncomp(FiltBSGAAI, 4);
    ConCompFBSGAAI.NumObjects;
    ImageData = regionprops(ConCompFBSGAAI, 'basic');
    Areas = [ImageData.Area];
    [MaxSparkArea, idx] = max(Areas);
    if size (MaxSparkArea) == 0
    RawArea(k)= 0;
    else
        RawArea(k) = MaxSparkArea;
    end
    if IntensityAIFFMASSI>=(RawIntensity(k-1)*0.8) && IntensityAIFFMASSI>=↙
(FinalIntensity(m-1)*0.8)
            FinalIntensity(m) = IntensityAIFFMASSI;
            FinalArea(m) = RawArea(k);
            m = m + 1;
    end

end


%%%%%%%%%%%%%%%%%%%%% Display of resulting graphs %%%%%%%%%%%%%%%%%%%%%%%%%

Rawtime = linspace(5, TotalNumImages*5, TotalNumImages);
FinaltimeIntensity = linspace(5,(m-1)*5,m-1);
FinaltimeArea = linspace(5,(m-1)*5,m-1);
Insertstime = linspace(0,(m-1)*5,InsertIndex);
InsertPixSize = 12.6/590;
InsertPixArea = InsertPixSize*InsertPixSize;
InsertRealMaxArea = MaxArea*InsertPixArea;

plot(Insertstime,InsertRealMaxArea,'-*');
title('Insert Wear Area','FontSize',11);
xlabel('Time (s)','FontSize',10);
ylabel('Area (mm2)','FontSize',10);
```

```

figure;
subplot(2,2,1);
plot(Rawtime,RawIntensity);
title('Raw Spark Intensity Evolution','FontSize',11);
xlabel('Time (s)','FontSize',10);
ylabel('Intensity Metric','FontSize',10);
subplot(2,2,2);
plot(FinaltimeIntensity,FinalIntensity);
title('Final Spark Intensity Evolution','FontSize',11);
xlabel('Time (s)','FontSize',10);
ylabel('Intensity Metric','FontSize',10);
subplot(2,2,3);
plot(Rawtime,RawArea);
title('Raw Spark Area Evolution','FontSize',11);
xlabel('Time (s)','FontSize',10);
ylabel('Area (Number of pixels)','FontSize',10);
subplot(2,2,4);
plot(FinaltimeArea,FinalArea);
title('Final Spark Area Evolution','FontSize',11);
xlabel('Time (s)','FontSize',10);
ylabel('Area (Number of pixels)','FontSize',10);

%The command "plotyyy" was downloaded from interent and developed by
%another MATLAB user.
ylabels{1}='Intensity Metric';
ylabels{2}='Area (Number of pixels)';
ylabels{3}='Area (mm2)';
[ax,hlines] = plotyyy(FinaltimeIntensity,FinalIntensity,FinaltimeArea,FinalArea,↙
Insertstime,InsertRealMaxArea,ylabels);
legend(hlines, 'Spark Intensity','Spark Area','Crater Wear Area',↙
2,'Location','northwest');
xlabel(ax(1),'Time (s)');
```

APPENDIX VII – ALGORITHM OF SECOND SET OF EXPERIMENTS

Area Extraction with Tool Wear Values

18/09/17 13:32   D:\...\Images Combination Area Test2 V2.m   1 of 9

```
clearvars; close all; clc;
% for CombWin = 150:10:600
% Spark Area extracton of Combined Images.
%clear all; close all; clc;
       %%%VARIABLES INITIALIZATION AND PRE-PROCESSING SECTION%%%
TotImages = 7201;
CombWin = 300;                                                     %↙
Combination window size (number of images to be combined).
Area = zeros(1,fix((TotImages/CombWin)*8));                          %↙
Initialization of variable that stores area results from each combined image. 192 is↙
the total amount of combined images.
CombImageNum = 1;                                                  %This↙
variable will be used as an index for each Combined Image, in order to store their↙
intensity values at the Intensity variable above.
Level = 0.06;
RefImage = imread('Test2Ref.jpg');                                 %↙
Reference Image taken from the first frames of the video feed to use it for background↙
elimination.
RefImage = im2double(RefImage);                                    %↙
Conversion of Reference Image varaible to "Double" for future subtraction to each↙
image.
MultiImageIndex = 1;
MultiImage = zeros(600,800,CombWin);

         %%%IMAGE PROCESSING AND FEATURE EXTRACTION SECTION%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C1P1 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9                                               %↙
Creation of filename from 1 to 9.
        INString = num2str(ImageNum);                              %↙
Convert Image Number to String.
        filename = strcat('Test2C1P1_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P1_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P1_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= 7201)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P1_',INString,'.JPG');
    end
    Image = imread(filename);                                      %Read↙
Image to be processed.
    Image = im2double(Image);                                      %↙
Conversion of Image to Double.
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);                                   %This↙
command takes the remainder of the divition of Image Number and the Combination↙
window.
```

```
    if Rem == 0                                                             %If↙
the remainder is 0, the number of combined images (CombWin) is met and can now be↙
processed.
        ImageAdj = imadjust(CombImage,[.15; .3]);                           %Image↙
Adjustment to increase ilumination levels.
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageBinary = im2bw(ImageCrop,Level);                               %Image↙
conversion into Binary, using threshold value of Level.
        SARImage = bwareaopen(ImageBinary, 50);↙
%Small Areas Removal of objects smaller than 50 pixels.
        ConnComp = bwconncomp(SARImage, 4);                                 %↙
Command to find and store Connected Components.
        ImageProps = regionprops(ConnComp, 'basic');                        %Image↙
properties extraction of "basic" properties: Area, Centroids and BoundingBoxes.
        ImageAreas = [ImageProps.Area];                                     %↙
Extraction of only Area values from image properties.
        [MaxImageArea, idx] = max(ImageAreas);                              %↙
Selectionof Maximum Image Area and idx (index location vector).
        if size(MaxImageArea) == 0
            Area(CombImageNum) = 0;
        else
            Area(CombImageNum) = MaxImageArea;
        end
        CombImageNum = CombImageNum + 1;                                    %↙
Increase Combined Image Number by one each time the "if" is cleared to only store↙
combined image's intensity values.
        MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C1P2 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P2_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P2_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P2_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= 7201)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P2_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
```

18/09/17 13:32 D:\...\Images Combination Area Test2 V2.m   3 of 9

```
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.15; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageBinary = im2bw(ImageCrop,Level);                                   %Image↙
conversion into Binary, using threshold value of Level.
        SARImage = bwareaopen(ImageBinary, 50);↙
%Small Areas Removal of objects smaller than 50 pixels.
        ConnComp = bwconncomp(SARImage, 4);                                     %↙
Command to find and store Connected Components.
        ImageProps = regionprops(ConnComp, 'basic');                            %Image↙
properties extraction of "basic" properties: Area, Centroids and BoundingBoxes.
        ImageAreas = [ImageProps.Area];                                         %↙
Extraction of only Area values from image properties.
        [MaxImageArea, idx] = max(ImageAreas);                                  %↙
Selectionof Maximum Image Area and idx (index location vector).
        if size(MaxImageArea) == 0
            Area(CombImageNum) = 0;
        else
            Area(CombImageNum) = MaxImageArea;
        end
        CombImageNum = CombImageNum + 1;
        MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C1P3 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P3_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P3_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P3_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= 7201)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P3_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.15; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageBinary = im2bw(ImageCrop,Level);                               %Image↙
conversion into Binary, using threshold value of Level.
        SARImage = bwareaopen(ImageBinary, 50);↙
%Small Areas Removal of objects smaller than 50 pixels.
        ConnComp = bwconncomp(SARImage, 4);                                 %↙
Command to find and store Connected Components.
        ImageProps = regionprops(ConnComp, 'basic');                        %Image↙
properties extraction of "basic" properties: Area, Centroids and BoundingBoxes.
        ImageAreas = [ImageProps.Area];                                     %↙
Extraction of only Area values from image properties.
        [MaxImageArea, idx] = max(ImageAreas);                              %↙
Selectionof Maximum Image Area and idx (index location vector).
        if size(MaxImageArea) == 0
            Area(CombImageNum) = 0;
        else
            Area(CombImageNum) = MaxImageArea;
        end
        CombImageNum = CombImageNum + 1;
    MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C1P4 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P4_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P4_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P4_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= 7201)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P4_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.15; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageBinary = im2bw(ImageCrop,Level);                               %Image↙
conversion into Binary, using threshold value of Level.
        SARImage = bwareaopen(ImageBinary, 50);↙
%Small Areas Removal of objects smaller than 50 pixels.
```

```
18/09/17 13:32  D:\...\Images Combination Area Test2 V2.m   5 of 9

        ConnComp = bwconncomp(SARImage, 4);                                 %↙
Command to find and store Connected Components.
        ImageProps = regionprops(ConnComp, 'basic');                        %Image↙
properties extraction of "basic" properties: Area, Centroids and BoundingBoxes.
        ImageAreas = [ImageProps.Area];                                     %↙
Extraction of only Area values from image properties.
        [MaxImageArea, idx] = max(ImageAreas);                              %↙
Selectionof Maximum Image Area and idx (index location vector).
        if size(MaxImageArea) == 0
            Area(CombImageNum) = 0;
        else
            Area(CombImageNum) = MaxImageArea;
        end
        CombImageNum = CombImageNum + 1;
    MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C1P5 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P5_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P5_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P5_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= 7201)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P5_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.15; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageBinary = im2bw(ImageCrop,Level);                               %Image↙
conversion into Binary, using threshold value of Level.
        SARImage = bwareaopen(ImageBinary, 50);↙
%Small Areas Removal of objects smaller than 50 pixels.
        ConnComp = bwconncomp(SARImage, 4);                                 %↙
Command to find and store Connected Components.
        ImageProps = regionprops(ConnComp, 'basic');                        %Image↙
properties extraction of "basic" properties: Area, Centroids and BoundingBoxes.
```

```
        ImageAreas = [ImageProps.Area];                                    %↙
Extraction of only Area values from image properties.
        [MaxImageArea, idx] = max(ImageAreas);                             %↙
Selectionof Maximum Image Area and idx (index location vector).
        if size(MaxImageArea) == 0
            Area(CombImageNum) = 0;
        else
            Area(CombImageNum) = MaxImageArea;
        end
        CombImageNum = CombImageNum + 1;
    MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C2P1 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P1_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P1_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P1_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= 7201)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P1_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.15; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageBinary = im2bw(ImageCrop,Level);                              %Image↙
conversion into Binary, using threshold value of Level.
        SARImage = bwareaopen(ImageBinary, 50);↙
%Small Areas Removal of objects smaller than 50 pixels.
        ConnComp = bwconncomp(SARImage, 4);                                %↙
Command to find and store Connected Components.
        ImageProps = regionprops(ConnComp, 'basic');                       %Image↙
properties extraction of "basic" properties: Area, Centroids and BoundingBoxes.
        ImageAreas = [ImageProps.Area];                                    %↙
Extraction of only Area values from image properties.
        [MaxImageArea, idx] = max(ImageAreas);                             %↙
Selectionof Maximum Image Area and idx (index location vector).
```

18/09/17 13:32 D:\...\Images Combination Area Test2 V2.m   7 of 9

```
        if size(MaxImageArea) == 0
            Area(CombImageNum) = 0;
        else
            Area(CombImageNum) = MaxImageArea;
        end
        CombImageNum = CombImageNum + 1;
    MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C2P2 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P2_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P2_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P2_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= 7201)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P2_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.15; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageBinary = im2bw(ImageCrop,Level);                              %Image↙
conversion into Binary, using threshold value of Level.
        SARImage = bwareaopen(ImageBinary, 50);↙
%Small Areas Removal of objects smaller than 50 pixels.
        ConnComp = bwconncomp(SARImage, 4);                                %↙
Command to find and store Connected Components.
        ImageProps = regionprops(ConnComp, 'basic');                       %Image↙
properties extraction of "basic" properties: Area, Centroids and BoundingBoxes.
        ImageAreas = [ImageProps.Area];                                    %↙
Extraction of only Area values from image properties.
        [MaxImageArea, idx] = max(ImageAreas);                             %↙
Selectionof Maximum Image Area and idx (index location vector).
        if size(MaxImageArea) == 0
            Area(CombImageNum) = 0;
        else
            Area(CombImageNum) = MaxImageArea;
        end
        CombImageNum = CombImageNum + 1;
    MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

FirstImageIndex = 1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C2P3 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P3_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P3_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P3_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= 7201)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P3_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.15; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageBinary = im2bw(ImageCrop,Level);                              %Image↙
conversion into Binary, using threshold value of Level.
        SARImage = bwareaopen(ImageBinary, 50);↙
%Small Areas Removal of objects smaller than 50 pixels.
        ConnComp = bwconncomp(SARImage, 4);                                %↙
Command to find and store Connected Components.
        ImageProps = regionprops(ConnComp, 'basic');                       %Image↙
properties extraction of "basic" properties: Area, Centroids and BoundingBoxes.
        ImageAreas = [ImageProps.Area];                                    %↙
Extraction of only Area values from image properties.
        [MaxImageArea, idx] = max(ImageAreas);                             %↙
Selectionof Maximum Image Area and idx (index location vector).
        if size(MaxImageArea) == 0
            Area(CombImageNum) = 0;
        else
            Area(CombImageNum) = MaxImageArea;
        end
        CombImageNum = CombImageNum + 1;
    MultiImageIndex = 1;
    end
end

                         %%%REPORT SECTION%%%

plot(Area,'b');
xlabel('Time (s)','FontSize',14);
ylabel('Area (Num of pixels)','FontSize',14);

Time = (1:192);
Mass = [0 -0.00004 -0.00006 0.00265 0.00254 0.00470 0.00449 0.00614 0.00678];
MassTime = [0 24 48 72 96 120 144 168 192];

figure;
plot(MassTime,Mass,'r');
xlabel('Time (s)','FontSize',14);
ylabel('Mass (grams)','FontSize',14);

figure;
plot(MassTime,MaxAreamm,'r');
xlabel('Time (s)','FontSize',14);
ylabel('Area (Number of pixels)','FontSize',14);

figure;
ylabels{1}='Area (Number of pixels)';
ylabels{2}='Mass (grams)';
ylabels{3}='Area (Number of pixels)';
[ax,hlines] = plotyyy(Time,Area,MassTime,Mass,MassTime,MaxAreamm,ylabels);
legend(hlines, 'Spark Area','Mass Loss','Crater Wear Area',↙
2,'Location','northwest');
xlabel(ax(1),'Time (s)');
```

Intensity Extraction

```
18/09/17 13:41  D:\MATLA...\Images Combination Test2 V2.m   1 of 7
```

```
% Spark Intensity extracton of Combined Images with Low-Pass filter.
clear all; close all; clc;
       %%%VARIABLES INITIALIZATION AND PRE-PROCESSING SECTION%%%
Intensity = zeros(1,192);                                          % Initialization of variable that stores intensity results from each combined image. 192 is the total amount of combined images.
LPF = zeros(251,771);                                              % Initialization with zeros of "Low-Pass Filter" image of size 800 x 600 pixels.
FSize = 10;                                                        % Filter size (size of the filter square side).
LPF(125-(FSize/2):125+(FSize/2),385-(FSize/2):385+(FSize/2))= 1;   %This line ceates a square with values of ones in the filter image, making it a Low-Pass filter.
CombWin = 300;                                                     % Combination window size (number of images to be combined).
CombImageNum = 1;                                                  %This variable will be used as an index for each Combined Image, in order to store their intensity values at the Intensity variable above.
RefImage = imread('Test2Ref.jpg');                                 % Reference Image taken from the first frames of the video feed to use it for background elimination.
RefImage = im2double(RefImage);                                    % Conversion of Reference Image varaible to "Double" for future subtraction to each image.
TotImages = 7201;
MultiImageIndex = 1;
MultiImage = zeros(600,800,CombWin);

          %%%IMAGE PROCESSING AND FEATURE EXTRACTION SECTION%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C1P1 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9                                               % Creation of filename from 1 to 9.
        INString = num2str(ImageNum);                              % Convert Image Number to String.
        filename = strcat('Test2C1P1_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P1_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P1_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= TotImages)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P1_',INString,'.JPG');
    end
    Image = imread(filename);                                      %Read Image to be processed.
    Image = im2double(Image);                                      % Conversion of Image to Double.
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
```

18/09/17 13:41 D:\MATLA...\Images Combination Test2 V2.m 2 of 7

```
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);                                            %This↙
command takes the remainder of the divition of Image Number and the Combination↙
window.
    if Rem == 0                                                             %If↙
the remainder is 0, the number of combined images (CombWin) is met and can now be↙
processed.
        ImageAdj = imadjust(CombImage,[.25; .3]);                           %Image↙
Adjustment to increase ilumination levels.
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageFreq = fftshift(fft2(ImageCrop));                              %↙
Conversion of Image to frequency domain.
        LPFImageFreq = ImageFreq.*LPF;                                      %↙
Application of the Low-Pass filter.
        LPFImageRaw = ifft2(ifftshift(LPFImageFreq));                       %↙
Conversion of Low-Pass Filtered Image with raw (there are positive and negative↙
values) data from frequency domain.
        LPFImage = abs(LPFImageRaw);                                        %Take↙
absolute values from LPFImageRaw to eliminate positive and negative values.
        Intensity(CombImageNum) = sum(sum(LPFImage));                       %Take↙
a total value of raw intensity in the image, where the spark is dominant.
        subplot(4,6,CombImageNum);
        imshow(CombImage);
        CombImageNum = CombImageNum + 1;                                    %↙
Increase Combined Image Number by one each time the "if" is cleared to only store↙
combined image's intensity values.
        MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C1P2 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P2_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P2_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P2_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= TotImages)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P2_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.25; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageFreq = fftshift(fft2(ImageCrop));
        LPFImageFreq = ImageFreq.*LPF;
        LPFImageRaw = ifft2(ifftshift(LPFImageFreq));
        LPFImage = abs(LPFImageRaw);
        Intensity(CombImageNum) = sum(sum(LPFImage));
        CombImageNum = CombImageNum + 1;
        MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C1P3 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P3_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P3_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P3_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= TotImages)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P3_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.25; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageFreq = fftshift(fft2(ImageCrop));
        LPFImageFreq = ImageFreq.*LPF;
        LPFImageRaw = ifft2(ifftshift(LPFImageFreq));
        LPFImage = abs(LPFImageRaw);
        Intensity(CombImageNum) = sum(sum(LPFImage));
        CombImageNum = CombImageNum + 1;
        MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

```

```
18/09/17 13:41  D:\MATLA...\Images Combination Test2 V2.m   4 of 7

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C1P4 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P4_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P4_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P4_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= TotImages)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P4_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.25; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageFreq = fftshift(fft2(ImageCrop));
        LPFImageFreq = ImageFreq.*LPF;
        LPFImageRaw = ifft2(ifftshift(LPFImageFreq));
        LPFImage = abs(LPFImageRaw);
        Intensity(CombImageNum) = sum(sum(LPFImage));
        CombImageNum = CombImageNum + 1;
        MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C1P5 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P5_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P5_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P5_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= TotImages)
        INString = num2str(ImageNum);
        filename = strcat('Test2C1P5_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.25; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageFreq = fftshift(fft2(ImageCrop));
        LPFImageFreq = ImageFreq.*LPF;
        LPFImageRaw = ifft2(ifftshift(LPFImageFreq));
        LPFImage = abs(LPFImageRaw);
        Intensity(CombImageNum) = sum(sum(LPFImage));
        CombImageNum = CombImageNum + 1;
        MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C2P1 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P1_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P1_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P1_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= TotImages)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P1_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.25; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageFreq = fftshift(fft2(ImageCrop));
        LPFImageFreq = ImageFreq.*LPF;
        LPFImageRaw = ifft2(ifftshift(LPFImageFreq));
        LPFImage = abs(LPFImageRaw);
        Intensity(CombImageNum) = sum(sum(LPFImage));
        CombImageNum = CombImageNum + 1;
        MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C2P2 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P2_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P2_00',INString,'.JPG');
    end
    if (100 <= ImageNum) && (ImageNum <= 999)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P2_0',INString,'.JPG');
    end
    if (1000 <= ImageNum) && (ImageNum <= TotImages)
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P2_',INString,'.JPG');
    end
    Image = imread(filename);
    Image = im2double(Image);
    MultiImage(:,:,MultiImageIndex) = Image;
    MultiImageIndex = MultiImageIndex + 1;
    CombImage = mean(MultiImage,3);
    Rem = rem(ImageNum,CombWin);
    if Rem == 0
        ImageAdj = imadjust(CombImage,[.25; .3]);
        ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
        ImageFreq = fftshift(fft2(ImageCrop));
        LPFImageFreq = ImageFreq.*LPF;
        LPFImageRaw = ifft2(ifftshift(LPFImageFreq));
        LPFImage = abs(LPFImageRaw);
        Intensity(CombImageNum) = sum(sum(LPFImage));
        CombImageNum = CombImageNum + 1;
        MultiImageIndex = 1;
    end
end
clear ImageNum INString filename Image FirstImageIndex CombImage R ImageAdj
clear ImageFreq LPFImageFreq LPFImageRaw LPFImage

FirstImageIndex = 1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Test2C2P3 images%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ImageNum = 1:TotImages
    if ImageNum <= 9
        INString = num2str(ImageNum);
        filename = strcat('Test2C2P3_000',INString,'.JPG');
    end
    if (10 <= ImageNum) && (ImageNum <= 99)
        INString = num2str(ImageNum);
```

```
            filename = strcat('Test2C2P3_00',INString,'.JPG');
        end
        if (100 <= ImageNum) && (ImageNum <= 999)
            INString = num2str(ImageNum);
            filename = strcat('Test2C2P3_0',INString,'.JPG');
        end
        if (1000 <= ImageNum) && (ImageNum <= TotImages)
            INString = num2str(ImageNum);
            filename = strcat('Test2C2P3_',INString,'.JPG');
        end
        Image = imread(filename);
        Image = im2double(Image);
        MultiImage(:,:,MultiImageIndex) = Image;
        MultiImageIndex = MultiImageIndex + 1;
        CombImage = mean(MultiImage,3);
        Rem = rem(ImageNum,CombWin);
        if Rem == 0
            ImageAdj = imadjust(CombImage,[.25; .3]);
            ImageCrop = imcrop(ImageAdj,[30 350 770 250]);
            ImageFreq = fftshift(fft2(ImageCrop));
            LPFImageFreq = ImageFreq.*LPF;
            LPFImageRaw = ifft2(ifftshift(LPFImageFreq));
            LPFImage = abs(LPFImageRaw);
            Intensity(CombImageNum) = sum(sum(LPFImage));
            CombImageNum = CombImageNum + 1;
            MultiImageIndex = 1;
        end
    end

                            %%%REPORT SECTION%%%

    plot(Intensity,'b');
    xlabel('Time (s)','FontSize',14);
    ylabel('Intensity Metric','FontSize',14);
```

APPENDIX VIII – ALGORITHM OF THIRD SET OF EXPERIMENTS

Example of Test 3.

```
18/09/17 13:45  D:...\Spark Area and Intensity Test3 V2.m   1 of 4

clearvars, clc, close all;
%This programme includes a manual value of 20 images per run (see the
%filtering section after area calculation).
prompt = 'What is the initial number in the video file name? (e.g. for MVI_0103.↙
mov, write = 103)   ';
Initial = input(prompt);
prompt = 'What is the last number in the video file name? (e.g. for MVI_0130.mov,↙
write = 130)   ';
Final = input(prompt);
NumVideos = (Final - Initial) + 1;
Intensity = cast([], 'double');
Area = cast([], 'double');
AreaFilt = cast([], 'double');
IntensityFilt = cast([], 'double');
TestIntensity = cast([], 'double');
TestArea = cast([], 'double');
IntIndex = 1;
AreaIndex = 1;
RunFrames = 26;
for v = 1:NumVideos
    filename = sprintf('MVI_%04d.mov', (Initial-1)+v);

    %Convert frames of video to images
    vidObj = VideoReader(filename);                                      %Read↙
video (change filename for each video).
    vidHeight = vidObj.Height;                                           %↙
Determine the height and width of the frame.
    vidWidth = vidObj.Width;
    Vid = struct('cdata',zeros(vidHeight,vidWidth,3,'uint8'));           %Create↙
a MATLAB structure array for movie frames in uint8 (0 to 255.
    k = 1;
    while hasFrame(vidObj)                                               %Read↙
all frames until the end; variable "cdata" contains each frame.
        Vid(k).cdata = readFrame(vidObj);
        k = k+1;
    end

    %Background subtraction
    Isubt = struct('SubtractI',zeros(vidHeight,vidWidth,3,'uint8'));     %↙
Structure variables creation.
    AdjI = struct('AdjustedI',zeros(vidHeight,vidWidth,3,'uint8'));
    NumFrames = size(Vid,2);                                             %↙
Extract the number of frames in video.
    for i=1:NumFrames
        G = Vid(i).cdata(:,:,2);                                         %↙
Separate Green channel of frame.
        B = Vid(i).cdata(:,:,3);                                         %↙
Saparate Blue channel of frame.
        se = strel('disk',2,8);                                          %Create↙
morphological structure element in the from of a disk with radius of 2 pixels.
        BackgroundG = imopen(G,se);                                      %↙
Morphologically open image using morphological disk structures to create backgrounds.
        BackgroundB = imopen(B,se);
        Isubt(i).SubtractI = imsubtract(Vid(i).cdata, cat(3, BackgroundB,↙
BackgroundG, BackgroundB));    %Subtract background, created from open image and using↙
```

```
the Blue channel in the Red channel.
        AdjI(i).AdjustedI = imadjust(Isubt(i).SubtractI,[.01 .2 .01; 1 .5 .011],[0↙
0 0; 1 0.001 .001]);  %Enhance Image to isolate spark.
    end

    %Image combination
    WindowSize = vidObj.FrameRate;                                          %↙
Calculate the combination window size using the frame rate.
    Icomb = struct('CombI',zeros(vidHeight,vidWidth,3,'uint8'));            %Cast↙
structure variable for combined images.
    I = im2uint8(zeros(vidHeight,vidWidth,3));
    k = 1;
    for i=1:NumFrames                                                       %Image↙
Combination into new variable Icomb.
        I = AdjI(i).AdjustedI + I;
        Rem = rem(i,WindowSize);
        if Rem == 0
            Icomb(k).CombI = I;
            I = im2uint8(zeros(vidHeight,vidWidth,3));
            k = k + 1;
        end
    end

    NumImages = size(Icomb,2);
    SparkIndex = cast([], 'double');
    for i = 1:NumImages
        %%%%Spark Intensity Extraction%%%%
        ImageGray = rgb2gray(Icomb(i).CombI);
        ImageFreq = fftshift(fft2(ImageGray));
        %Filter size calculation
        ImageFreqAbs = abs(ImageFreq);
        CentreX = round(vidWidth/2)+1;
        x = CentreX;
        CentreY = round(vidHeight/2)+1;
        y = CentreY;
        if ImageFreqAbs(CentreY,CentreX)>200
            while ImageFreqAbs(y,x)>=(0.7*ImageFreqAbs(CentreY,CentreX))
                x = x + 1;
            end
            xx = CentreX;
            while ImageFreqAbs(y,xx)>=(0.7*ImageFreqAbs(CentreY,CentreX))
                y = y + 1;
            end
            %Filter Implementation
            LPF = zeros(vidHeight,vidWidth);
            LPF(CentreY-(y-CentreY):CentreY+(y-CentreY),CentreX-(x-CentreX):↙
CentreX+(x-CentreX))= 1;
            LPFImageFreq = ImageFreq.*LPF;
            LPFImageRaw = ifft2(ifftshift(LPFImageFreq));
            LPFImage = abs(LPFImageRaw);
            Intensity(IntIndex) = sum(sum(LPFImage));
            TestIntensity(i) = sum(sum(LPFImage));
        else
            Intensity(IntIndex) = 0;
            TestIntensity(i) = 0;
```

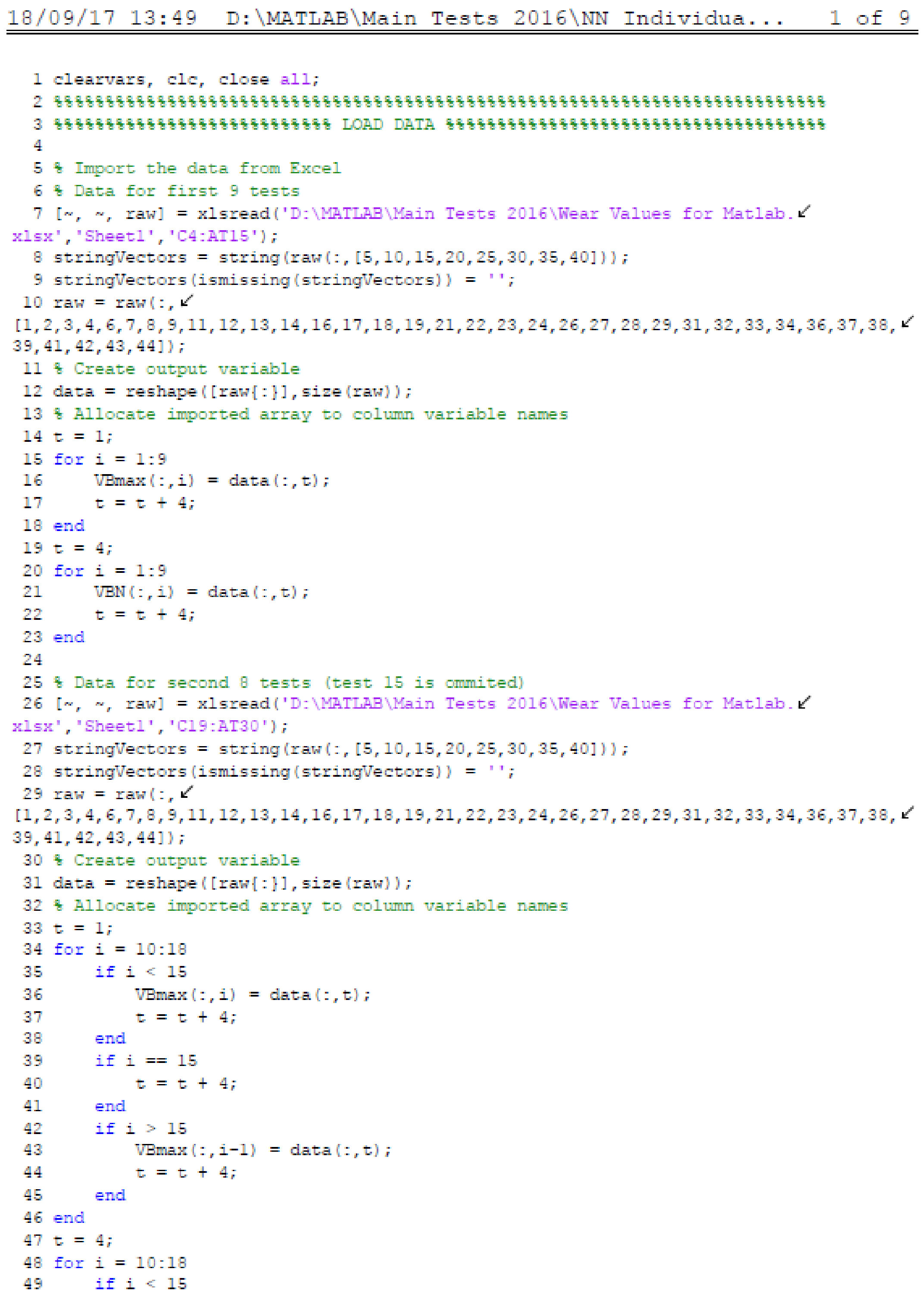
18/09/17 13:45 D:...\Spark Area and Intensity Test3 V2.m   3 of 4

```
        end
        IntIndex = IntIndex + 1;

        %%%%Spark Area Extraction%%%%
        ImageBinary = im2bw(Icomb(i).CombI,0.06);
        SARImage = bwareaopen(ImageBinary, 50);                           %Small↙
Areas Removal of objects smaller than 50 pixels.
        ConnComp = bwconncomp(SARImage, 4);                               %Find↙
and store Connected Components.
        ImageProps = regionprops(ConnComp, 'basic');                      %Image↙
properties extraction of "basic" properties: Area, Centroids and BoundingBoxes.
        ImageAreas = [ImageProps.Area];                                   %↙
Extraction of only Area values from image properties.
        [MaxImageArea, idx] = max(ImageAreas);
%          Spark = false(size(ImageBinary));                 i            %↙
Create image named Spark that only shows the binarised spark.
%          Spark(ConnComp.PixelIdxList{idx}) = true;
        if size(MaxImageArea) == 0
            Area(AreaIndex) = 0;
            TestArea(i) = 0;
        else
            Area(AreaIndex) = MaxImageArea;
            TestArea(i) = MaxImageArea;
        end
        AreaIndex = AreaIndex + 1;
    end

    Size = size(AreaFilt,2);
    AreaStd = std(TestArea);
    Loc = find(TestArea>(AreaStd*0.8),1);
    if Loc > floor(size(TestArea,2)/2)
        AreaFilt(Size+1:Size+RunFrames+1) = TestArea(size(TestArea,2)-RunFrames:↙
size(TestArea,2));
    else
        AreaFilt(Size+1:Size+RunFrames+1) = TestArea(Loc:Loc+RunFrames);
    end

    Size = size(IntensityFilt,2);
    IntensityStd = std(TestIntensity);
    Loc = find(TestIntensity>(IntensityStd*0.8),1);
    if Loc > floor(size(TestIntensity,2)/2)
        IntensityFilt(Size+1:Size+RunFrames+1) = TestIntensity(size(TestIntensity,↙
2)-RunFrames:size(TestIntensity,2));
    else
        IntensityFilt(Size+1:Size+RunFrames+1) = TestIntensity(Loc:Loc+RunFrames);
    end
end

AreaFilt(2:size(AreaFilt,2)+1) = AreaFilt(:);
AreaFilt(1) = 0;
IntensityFilt(2:size(IntensityFilt,2)+1) = IntensityFilt(:);
IntensityFilt(1) = 0;

figure;
subplot(3,2,1);
```

```
plot(Area,'b');
xlabel('Time (s)','FontSize',14);
ylabel('Area (Number of pixels)','FontSize',14);
subplot(3,2,2);
plot(Intensity,'r');
xlabel('Time (s)','FontSize',14);
ylabel('Intensity Metric','FontSize',14);


subplot(3,2,3);
plot(AreaFilt,'b');
xlabel('Time (s)','FontSize',14);
ylabel('Area (Number of pixels)','FontSize',14);
subplot(3,2,4);
plot(IntensityFilt,'r');
xlabel('Time (s)','FontSize',14);
ylabel('Intensity Metric','FontSize',14);

Area3 = zscore(AreaFilt);
Intensity3 = zscore(IntensityFilt);
subplot(3,2,5);
plot(Area3,'b');
xlabel('Time (s)','FontSize',14);
ylabel('Area (Number of pixels)','FontSize',14);
subplot(3,2,6);
plot(Intensity3,'r');
xlabel('Time (s)','FontSize',14);
ylabel('Intensity Metric','FontSize',14);

```

APPENDIX IX – INDIVIDUAL TESTS NETWORKS ALGORITHM

Example of Test 2 with Spark Event Input.



18/09/17 13:49 D:\MATLAB\Main Tests 2016\NN Individua... 1 of 9

```
clearvars, clc, close all;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% LOAD DATA %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Import the data from Excel
% Data for first 9 tests
[~, ~, raw] = xlsread('D:\MATLAB\Main Tests 2016\Wear Values for Matlab.↙
xlsx','Sheet1','C4:AT15');
stringVectors = string(raw(:,[5,10,15,20,25,30,35,40]));
stringVectors(ismissing(stringVectors)) = '';
raw = raw(:,↙
[1,2,3,4,6,7,8,9,11,12,13,14,16,17,18,19,21,22,23,24,26,27,28,29,31,32,33,34,36,37,38,↙
39,41,42,43,44]);
% Create output variable
data = reshape([raw{:}],size(raw));
% Allocate imported array to column variable names
t = 1;
for i = 1:9
    VBmax(:,i) = data(:,t);
    t = t + 4;
end
t = 4;
for i = 1:9
    VBN(:,i) = data(:,t);
    t = t + 4;
end

% Data for second 8 tests (test 15 is ommited)
[~, ~, raw] = xlsread('D:\MATLAB\Main Tests 2016\Wear Values for Matlab.↙
xlsx','Sheet1','C19:AT30');
stringVectors = string(raw(:,[5,10,15,20,25,30,35,40]));
stringVectors(ismissing(stringVectors)) = '';
raw = raw(:,↙
[1,2,3,4,6,7,8,9,11,12,13,14,16,17,18,19,21,22,23,24,26,27,28,29,31,32,33,34,36,37,38,↙
39,41,42,43,44]);
% Create output variable
data = reshape([raw{:}],size(raw));
% Allocate imported array to column variable names
t = 1;
for i = 10:18
    if i < 15
        VBmax(:,i) = data(:,t);
        t = t + 4;
    end
    if i == 15
        t = t + 4;
    end
    if i > 15
        VBmax(:,i-1) = data(:,t);
        t = t + 4;
    end
end
t = 4;
for i = 10:18
    if i < 15
```

```
        VBN(:,i) = data(:,t);
        t = t + 4;
    end
    if i == 15
        t = t + 4;
    end
    if i > 15
        VBN(:,i-1) = data(:,t);
        t = t + 4;
    end
end
VBmax = zscore(VBmax);
VBN = zscore(VBN);
% Clear temporary variables
clearvars data raw stringVectors;

% Load Area, Intensity and Crater Wear data
for i = 1:17
    filename = sprintf('Area3%02d.mat', i);
    Inputs(i).Area = importdata(filename)';
    filename = sprintf('Intensity3%02d.mat', i);
    Inputs(i).Int = importdata(filename)';
    filename = sprintf('MaxAreamm%02d.mat', i);
    Craters(2:12,i) = importdata(filename)';
    Craters(1,i) = 0;
end
Craters = zscore(Craters);
t = 0:11;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%% FIT CURVES & MACHINING PARAMETERS %%%%%%%%%%%%%%%%%%%%%%

%%Fit curves with linear interpolation and generate machining variables.
for i = 1:17
    s = size(Inputs(i).Area,1);
    t = 0:(s/11):s;
    t = t';
    Time(i).tt = [0:(s-1)]';
    fitobject = fit(t,VBN(:,i),'linearinterp');   %Use either cubicinterp or↙
linearinterp
    Outputs(i).VBN = feval(fitobject,Time(i).tt);
    fitobject = fit(t,VBmax(:,i),'linearinterp');
    Outputs(i).VBmax = feval(fitobject,Time(i).tt);
    fitobject = fit(t,Craters(:,i),'linearinterp');
    Outputs(i).Crater = feval(fitobject,Time(i).tt);
%     figure; plot(time(i).tt,Craters2(i).Crater,t,Craters(:,i));     %Use only to↙
evaluate fitting
%     figure; plot(time(i).tt,VBmax2(i).VBmax,t,VBmax(:,i));

    % Create normalised machining variables and load into the 'Input'.
    %Cutting Speed variables
    if i== 2 || i==3 || i==8 || i==9 || i==10
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).Cut = Value';
        clear Value;
    end
    if i== 4 || i==6 || i==12 || i==15 || i==16 || i==17
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).Cut = Value';
        clear Value;
    end
    if i== 1 || i==5 || i==7 || i==11 || i==13 || i==14
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).Cut = Value';
        clear Value;
    end

    %Feed variables
    if i== 5 || i==8 || i==9 || i==11 || i==12 || i==17
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).Feed = Value';
        clear Value;
    end
    if i== 1 || i==2 || i==3 || i==6 || i==13 || i==15
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).Feed = Value';
        clear Value;
    end
    if i== 4 || i==7 || i==10 || i==14 || i==16
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).Feed = Value';
        clear Value;
    end

    %Cutting Depth variables
    if i== 2 || i==6 || i==7 || i==8 || i==11 || i==16
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).Depth = Value';
        clear Value;
    end
    if i== 1 || i==9 || i==10 || i==12 || i==14 || i==15
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).Depth = Value';
        clear Value;
    end
    if i== 3 || i==4 || i==5 || i==13 || i==17
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).Depth = Value';
        clear Value;
    end

    %Radial Immersion variables
    if i== 6 || i==8 || i==10 || i==13 || i==14 || i==17
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).RadIm = Value';
        clear Value;
    end
    if i== 1 || i==2 || i==4 || i==11 || i==12
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).RadIm = Value';
        clear Value;
    end
    if i== 3 || i==5 || i==7 || i==9 || i==15 || i==16
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).RadIm = Value';
        clear Value;
    end
end

for i = 1:17
    InputsL(i).Area = Inputs(i).Area+10;
end

% Flaking event detection and input creation
per = 0.97;
for i = 1:17
    count = 0;
    s = size(InputsL(i).Area,1);
    SparkEvent = zeros(s,1);
    for k = 6:(s-4)
        if (InputsL(i).Area(k-5)*per)&&(InputsL(i).Area(k-4)*per)&&(InputsL(i).↙
Area(k-3)*per)&&...
                (InputsL(i).Area(k-2)*per)&&(InputsL(i).Area(k-1)*per) > ...
                (InputsL(i).Area(k))&&(InputsL(i).Area(k+1))&&(InputsL(i).Area↙
(k+2))&&...
                (InputsL(i).Area(k+3))&&InputsL(i).Area(k+4)
            SparkEvent(k:s,1) = (count + 1)/100;
            count = count + 1;
        end
    end
    Inputs(i).SE = SparkEvent;
```

```
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%% MATRICES FOR TRAINING, VALIDATION AND TESTING %%%%%%%%%%%%%%

% Randomise data points and re-assign for Training, Validation and Testing
% matrices of all each Test.
load('seed2.mat');
TestNum = 2;
s = size(Inputs(TestNum).Area,1);
rng(seed);
r = rand(s,1);
k = 1;
j = 1;
h = 1;
for i = 1:s
    if r(i) <= 0.8
        InTestTr(j,1) = Inputs(TestNum).Area(i);
        InTestTr(j,2) = Inputs(TestNum).Int(i);
        InTestTr(j,3) = Inputs(TestNum).SE(i);
        OuTestTr(j,1) = Outputs(TestNum).VBN(i);
        OuTestTr(j,2) = Outputs(TestNum).VBmax(i);
        OuTestTr(j,3) = Outputs(TestNum).Crater(i);
        j = j + 1;
    elseif 0.9 >= r(i) > 0.8
        InTestVal(k,1) = Inputs(TestNum).Area(i);
        InTestVal(k,2) = Inputs(TestNum).Int(i);
        InTestVal(k,3) = Inputs(TestNum).SE(i);
        OuTestVal(k,1) = Outputs(TestNum).VBN(i);
        OuTestVal(k,2) = Outputs(TestNum).VBmax(i);
        OuTestVal(k,3) = Outputs(TestNum).Crater(i);
        k = k + 1;
    else
        InTestTest(h,1) = Inputs(TestNum).Area(i);
        InTestTest(h,2) = Inputs(TestNum).Int(i);
        InTestTest(h,3) = Inputs(TestNum).SE(i);
        OuTestTest(h,1) = Outputs(TestNum).VBN(i);
        OuTestTest(h,2) = Outputs(TestNum).VBmax(i);
        OuTestTest(h,3) = Outputs(TestNum).Crater(i);
        h = h + 1;
    end
end
% seed = rng;   %Use it to record the random generator seed for repetition.

MaxWeights = ((s*.8)/6)-0.5;
display(MaxWeights);
prompt = 'What is the maximum number of hidden nodes for evaluation?   ';
MaxHN = input(prompt);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% NEURAL NETWORKS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Cross validation to get number of hidden units.
for o = 1:10
    rng(o);
    s = rng;
    for i = 1:MaxHN
        % Set up network parameters.
        nin = 3;              % Number of inputs.
        nhidden = i;              % Number of hidden units.
        nout = 3;             % Number of outputs.
        alpha = 0.01;             % Coefficient of weight-decay prior.

        % Create and initialize network weight vector.
        net = mlp(nin, nhidden, nout, 'linear', alpha);

        % Set up vector of options for the optimiser.
        options = zeros(1,18);
        options(1) = -1;              % This provides display of error values.
        options(14) = 1000;       % Number of training cycles.

        % Train using scaled conjugate gradients.
        [net, options, errlog] = netopt(net, options, InTestTr, OuTestTr, 'scg');↙
%Also use scg
        yTr = mlpfwd(net, InTestTr);
        [eTr, edataTr, epriorTr] = mlperr(net, InTestTr, OuTestTr);
        MSETr(i,o) = edataTr/(size(InTestTr,1)*1.5);

        yVal = mlpfwd(net, InTestVal);
        [eVal, edataVal, epriorVal] = mlperr(net, InTestVal, OuTestVal);
        MSEVal(i,o) = edataVal/(size(InTestVal,1)*1.5);

        yTest = mlpfwd(net, InTestTest);
        [eTest, edataTest, epriorTest] = mlperr(net, InTestTest, OuTestTest);
        MSETest(i,o) = edataTest/(size(InTestTest,1)*1.5);
    end
end

[M,I] = min(MSEVal);
Minimums(1,:) = M;
Minimums(2,:) = I;
display(Minimums);

prompt = 'What is the best NN number [1 10]?   ';
NN = input(prompt);
prompt = 'What is the best number of nidden nodes?   ';
HN = input(prompt);

for o = 1:NN
    rng(o);
    s = rng;
    for i = 1:HN
        % Set up network parameters.
        nin = 3;              % Number of inputs.
        nhidden = i;              % Number of hidden units.
        nout = 3;             % Number of outputs.
        alpha = 0.01;             % Coefficient of weight-decay prior.

        % Create and initialize network weight vector.
        net = mlp(nin, nhidden, nout, 'linear', alpha);

        % Set up vector of options for the optimiser.
        options = zeros(1,18);
        options(1) = -1;            % This provides display of error values.
        options(14) = 1000;     % Number of training cycles.

        % Train using scaled conjugate gradients.
        [net, options, errlog2] = netopt(net, options, InTestTr, OuTestTr, 'scg');↙
%Also use scg
        yTr = mlpfwd(net, InTestTr);
        [eTr, edataTr, epriorTr] = mlperr(net, InTestTr, OuTestTr);
        MSETr2(i,o) = edataTr/(size(InTestTr,1)*1.5);

        yVal = mlpfwd(net, InTestVal);
        [eVal, edataVal, epriorVal] = mlperr(net, InTestVal, OuTestVal);
        MSEVal2(i,o) = edataVal/(size(InTestVal,1)*1.5);

        yTest = mlpfwd(net, InTestTest);
        [eTest, edataTest, epriorTest] = mlperr(net, InTestTest, OuTestTest);
        MSETest2(i,o) = edataTest/(size(InTestTest,1)*1.5);
    end
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%% RESULTS GRAPHING %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
x = 1:MaxHN;
Mean = mean(MSETr,2);
figure;
plot(x,MSETr,'.',x,Mean,'--','MarkerSize',7);
% title({'Training MSE for different number of hidden nodes.'},'FontSize',14);
legend('NN 1','NN 2','NN 3','NN 4','NN 5','NN 6','NN 7','NN 8','NN 9','NN↙
10','Mean Values');
xlabel('Number of Hidden Nodes','FontSize',14);
ylabel('Mean-Squared Error','FontSize',14);
Mean = mean(MSEVal,2);
figure;
plot(x,MSEVal,'.',x,Mean,'--','MarkerSize',7);
% title({'Validation MSE for different number of hidden nodes.'},'FontSize',14);
legend('NN 1','NN 2','NN 3','NN 4','NN 5','NN 6','NN 7','NN 8','NN 9','NN↙
10','Mean Values');
xlabel('Number of Hidden Nodes','FontSize',14);
ylabel('Mean-Squared Error','FontSize',14);

figure;
plot(MSEVal(:,NN));
% title({'Validation MSE values for different number of hidden nodes.'; 'NN number↙
9.'},'FontSize',14);
xlabel('Number of Hidden Nodes','FontSize',14);
ylabel('Mean-Squared Error','FontSize',14);

figure;
plot(errlog2);
title({'Bayesian training error values for 1000 iterations.'; 'NN number 9 with 13↙
Hidden Nodes.'},'FontSize',14);
xlabel('Iterations','FontSize',14);
```

```
ylabel('Bayesian Error','FontSize',14);

figure;
plot(yTr(:,1));
hold on;
plot(OuTestTr(:,1),'--');
% title({'Training and Prediction values of Notch Wear.'; 'NN number 9 with 13↙
Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Notch Wear','FontSize',14);
figure;
plot(yTr(:,2));
hold on;
plot(OuTestTr(:,2),'--');
% title({'Training and Prediction values of Flank Wear.'; 'NN number 9 with 13↙
Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Flank Wear','FontSize',14);
figure;
plot(yTr(:,3));
hold on;
plot(OuTestTr(:,3),'--');
% title({'Training and Prediction values of Crater Wear.'; 'NN number 9 with 13↙
Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Crater Wear','FontSize',14);

figure;
plot(yVal(:,1));
hold on;
plot(OuTestVal(:,1),'--');
title({'Validation and Prediction values of Notch Wear.'; 'NN number 9 with 13↙
Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Normalised Notch Wear','FontSize',14);
figure;
plot(yVal(:,2));
hold on;
plot(OuTestVal(:,2),'--');
title({'Validation and Prediction values of Flank Wear.'; 'NN number 9 with 13↙
Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Normalised Notch Wear','FontSize',14);
figure;
plot(yVal(:,3));
hold on;
plot(OuTestVal(:,3),'--');
title({'Validation and Prediction values of Crater Wear.'; 'NN number 9 with 13↙
Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
```

```
xlabel('Data Points','FontSize',14);
ylabel('Normalised Notch Wear','FontSize',14);

figure;
plot(yTest(:,1));
hold on;
plot(OuTestTest(:,1),'--');
title({'Testing and Prediction values of Notch Wear.'; 'NN number 9 with 13 Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Normalised Notch Wear','FontSize',14);
figure;
plot(yTest(:,2));
hold on;
plot(OuTestTest(:,2),'--');
title({'Testing and Prediction values of Flank Wear.'; 'NN number 9 with 13 Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Normalised Notch Wear','FontSize',14);
figure;
plot(yTest(:,3));
hold on;
plot(OuTestTest(:,3),'--');
title({'Testing and Prediction values of Crater Wear.'; 'NN number 9 with 13 Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Normalised Notch Wear','FontSize',14);

figure;
plot(InTestTr);
legend('Spark Area','Spark Intensity','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Regularised Spark Feature','FontSize',14);


```

APPENDIX X – RANDOMISED DATA NETWORK ALGORITHM WITH SPARK EVENT

18/09/17 13:53 D:\MATLA...\NN Tests CrossValidation SE.m   1 of 8

```
clearvars, clc, close all;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%% LOAD DATA %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Import the data from Excel
% Data for first 9 tests
[~, ~, raw] = xlsread('D:\MATLAB\Main Tests 2016\Wear Values for Matlab.↙
xlsx','Sheet1','C4:AT15');
stringVectors = string(raw(:,[5,10,15,20,25,30,35,40]));
stringVectors(ismissing(stringVectors)) = '';
raw = raw(:,↙
[1,2,3,4,6,7,8,9,11,12,13,14,16,17,18,19,21,22,23,24,26,27,28,29,31,32,33,34,36,37,38,↙
39,41,42,43,44]);
% Create output variable
data = reshape([raw{:}],size(raw));
% Allocate imported array to column variable names
t = 1;
for i = 1:9
    VBmax(:,i) = data(:,t);
    t = t + 4;
end
t = 4;
for i = 1:9
    VBN(:,i) = data(:,t);
    t = t + 4;
end

% Data for second 8 tests (test 15 is ommited)
[~, ~, raw] = xlsread('D:\MATLAB\Main Tests 2016\Wear Values for Matlab.↙
xlsx','Sheet1','C19:AT30');
stringVectors = string(raw(:,[5,10,15,20,25,30,35,40]));
stringVectors(ismissing(stringVectors)) = '';
raw = raw(:,↙
[1,2,3,4,6,7,8,9,11,12,13,14,16,17,18,19,21,22,23,24,26,27,28,29,31,32,33,34,36,37,38,↙
39,41,42,43,44]);
% Create output variable
data = reshape([raw{:}],size(raw));
% Allocate imported array to column variable names
t = 1;
for i = 10:18
    if i < 15
        VBmax(:,i) = data(:,t);
        t = t + 4;
    end
    if i == 15
        t = t + 4;
    end
    if i > 15
        VBmax(:,i-1) = data(:,t);
        t = t + 4;
    end
end
t = 4;
for i = 10:18
    if i < 15
```

```
        VBN(:,i) = data(:,t);
        t = t + 4;
    end
    if i == 15
        t = t + 4;
    end
    if i > 15
        VBN(:,i-1) = data(:,t);
        t = t + 4;
    end
end
VBmax = zscore(VBmax);
VBN = zscore(VBN);
% Clear temporary variables
clearvars data raw stringVectors;

% Load Area, Intensity and Crater Wear data
for i = 1:17
    filename = sprintf('Area3%02d.mat', i);
    Inputs(i).Area = importdata(filename)';
    filename = sprintf('Intensity3%02d.mat', i);
    Inputs(i).Int = importdata(filename)';
    filename = sprintf('MaxAreamm%02d.mat', i);
    Craters(2:12,i) = importdata(filename)';
    Craters(1,i) = 0;
end
Craters = zscore(Craters);
t = 0:11;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%% FIT CURVES & MACHINING PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%

%%Fit curves with linear interpolation and generate machining variables.
for i = 1:17
    s = size(Inputs(i).Area,1);
    t = 0:(s/11):s;
    t = t';
    Time(i).tt = [0:(s-1)]';
    fitobject = fit(t,VBN(:,i),'linearinterp');   %Use either cubicinterp or↙
linearinterp
    Outputs(i).VBN = feval(fitobject,Time(i).tt);
    fitobject = fit(t,VBmax(:,i),'linearinterp');
    Outputs(i).VBmax = feval(fitobject,Time(i).tt);
    fitobject = fit(t,Craters(:,i),'linearinterp');
    Outputs(i).Crater = feval(fitobject,Time(i).tt);
%     figure; plot(time(i).tt,Craters2(i).Crater,t,Craters(:,i));     %Use only to↙
evaluate fitting
%     figure; plot(time(i).tt,VBmax2(i).VBmax,t,VBmax(:,i));

    % Create normalised machining variables and load into the 'Input'.
    %Cutting Speed variables
    if i== 2 || i==3 || i==8 || i==9 || i==10
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).Cut = Value';
        clear Value;
    end
    if i== 4 || i==6 || i==12 || i==15 || i==16 || i==17
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).Cut = Value';
        clear Value;
    end
    if i== 1 || i==5 || i==7 || i==11 || i==13 || i==14
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).Cut = Value';
        clear Value;
    end

    %Feed variables
    if i== 5 || i==8 || i==9 || i==11 || i==12 || i==17
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).Feed = Value';
        clear Value;
    end
    if i== 1 || i==2 || i==3 || i==6 || i==13 || i==15
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).Feed = Value';
        clear Value;
    end
    if i== 4 || i==7 || i==10 || i==14 || i==16
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).Feed = Value';
        clear Value;
    end

    %Cutting Depth variables
    if i== 2 || i==6 || i==7 || i==8 || i==11 || i==16
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).Depth = Value';
        clear Value;
    end
    if i== 1 || i==9 || i==10 || i==12 || i==14 || i==15
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).Depth = Value';
        clear Value;
```

APPENDIX X – RANDOMISED DATA NETWORK ALGORITHM WITH SPARK EVENT

```
    end
    if i== 3 || i==4 || i==5 || i==13 || i==17
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).Depth = Value';
        clear Value;
    end

    %Radial Immersion variables
    if i== 6 || i==8 || i==10 || i==13 || i==14 || i==17
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).RadIm = Value';
        clear Value;
    end
    if i== 1 || i==2 || i==4 || i==11 || i==12
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).RadIm = Value';
        clear Value;
    end
    if i== 3 || i==5 || i==7 || i==9 || i==15 || i==16
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).RadIm = Value';
        clear Value;
    end
end

for i = 1:17
    InputsL(i).Area = Inputs(i).Area+10;
end

% Flaking event detection and input creation
per = 0.97;
for i = 1:17
    count = 0;
    s = size(InputsL(i).Area,1);
    SparkEvent = zeros(s,1);
    for k = 6:(s-4)
        if (InputsL(i).Area(k-5)*per)&&(InputsL(i).Area(k-4)*per)&&(InputsL(i).Area(k-3)*per)&&...
                (InputsL(i).Area(k-2)*per)&&(InputsL(i).Area(k-1)*per) > ...
                (InputsL(i).Area(k))&&(InputsL(i).Area(k+1))&&(InputsL(i).Area(k+2))&&...
                (InputsL(i).Area(k+3))&&InputsL(i).Area(k+4)
            SparkEvent(k:s,1) = (count + 1)/100;
            count = count + 1;
        end
    end
    Inputs(i).SE = SparkEvent;
```

```
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%% MATRICES FOR TRAINING, VALIDATION AND TESTING %%%%%%%%%%%%%%%
% Create matrix with collums correpsonding to the 6 inputs and
% columns to the time stamps of all tests CONCATENATED
s2 = 0;
for i = 1:17
    s = size(Inputs(i).Area,1);
    InputsTotMatCat(s2+1:s+s2,1) = Inputs(i).Area;
    s = size(Inputs(i).Int,1);
    InputsTotMatCat(s2+1:s+s2,2) = Inputs(i).Int;
    s = size(Inputs(i).Cut,1);
    InputsTotMatCat(s2+1:s+s2,3) = Inputs(i).Cut;
    s = size(Inputs(i).Feed,1);
    InputsTotMatCat(s2+1:s+s2,4) = Inputs(i).Feed;
    s = size(Inputs(i).Depth,1);
    InputsTotMatCat(s2+1:s+s2,5) = Inputs(i).Depth;
    s = size(Inputs(i).RadIm,1);
    InputsTotMatCat(s2+1:s+s2,6) = Inputs(i).RadIm;
    s = size(Inputs(i).SE,1);
    InputsTotMatCat(s2+1:s+s2,7) = Inputs(i).SE;
    s2 = size(InputsTotMatCat,1);
end

s2 = 0;
for i = 1:17
    s = size(Outputs(i).VBN,1);
    OutputsTotMatCat(s2+1:s+s2,1) = Outputs(i).VBN;
    k = k + 1;
    s = size(Outputs(i).VBmax,1);
    OutputsTotMatCat(s2+1:s+s2,2) = Outputs(i).VBmax;
    k = k + 1;
    s = size(Outputs(i).Crater,1);
    OutputsTotMatCat(s2+1:s+s2,3) = Outputs(i).Crater;
    k = k + 1;
    s2 = size(OutputsTotMatCat,1);
end

% Randomise data points and re-assign for Training, Validation and Testing
% matrices of all tests together.
load('seedALL.mat');
% seed = rng;   %Use it to record the random generator seed for repetition.
rng(seed);
r = rand(4428,1);
k = 1;
j = 1;
h = 1;
for i = 1:4428
    if r(i) <= 0.8
        InputsCVTrain(k,:) = InputsTotMatCat(i,:);
        OutputsCVTrain(k,:) = OutputsTotMatCat(i,:);
        k = k + 1;
    elseif 0.9 >= r(i) > 0.8
        InputsCVVal(j,:) = InputsTotMatCat(i,:);
```

18/09/17 13:53 D:\MATLA...\NN Tests CrossValidation SE.m   6 of 8

```matlab
        OutputsCVVal(j,:) = OutputsTotMatCat(i,:);
        j = j + 1;
    else
        InputsCVTest(h,:) = InputsTotMatCat(i,:);
        OutputsCVTest(h,:) = OutputsTotMatCat(i,:);
        h = h + 1;
    end
end

BestNodes = ((size(OutputsCVTrain,1)/8)-size(OutputsCVTrain,2))/(size↙
(InputsTotMatCat,2)+size(OutputsCVTrain,2)+1);
MinNodes = ((size(OutputsCVTrain,1)/3)-size(OutputsCVTrain,2))/(size↙
(InputsTotMatCat,2)+size(OutputsCVTrain,2)+1);
MidNodes = ((size(OutputsCVTrain,1)/2)-size(OutputsCVTrain,2))/(size↙
(InputsTotMatCat,2)+size(OutputsCVTrain,2)+1);
MaxNodes = ((size(OutputsCVTrain,1))-size(OutputsCVTrain,2))/(size↙
(InputsTotMatCat,2)+size(OutputsCVTrain,2)+1);
display(BestNodes);
display(MinNodes);
display(MidNodes);
display(MaxNodes);
prompt = 'What is the maximum number of hidden nodes for evaluation?   ';
MaxHN = input(prompt);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% NEURAL NETWORKS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Cross validation to get number of hidden units.
for o = 1:5
    rng(o);
    for i = 1:MaxHN
        ITNum = o + (i/1000);
        display(ITNum);
        % Set up network parameters.
        nin = 7;                % Number of inputs.
        nhidden = i;                % Number of hidden units.
        nout = 3;               % Number of outputs.
        alpha = 0.01;               % Coefficient of weight-decay prior.

        % Create and initialize network weight vector.
        net = mlp(nin, nhidden, nout, 'linear', alpha);

        % Set up vector of options for the optimiser.
        options = zeros(1,18);
        options(1) = -1;                % This provides display of error values.
        options(14) = 1000;     % Number of training cycles.

        % Train using scaled conjugate gradients.
        [net, options, errlog] = netopt(net, options, InputsCVTrain,↙
OutputsCVTrain, 'scg');  %Also use scg
        [eTr, edataTr, epriorTr] = mlperr(net, InputsCVTrain, OutputsCVTrain);
        MSETr(i,o) = edataTr/(size(InputsCVTrain,1)*1.5);

        yVal = mlpfwd(net, InputsCVVal);
        [eVal, edataVal, epriorVal] = mlperr(net, InputsCVVal, OutputsCVVal);
        MSEVal(i,o) = edataVal/(size(InputsCVVal,1)*1.5);

        yTest = mlpfwd(net, InputsCVTest);
        [eTest, edataTest, epriorTest] = mlperr(net, InputsCVTest, OutputsCVTest);
        MSETest(i,o) = edataTest/(size(InputsCVTest,1)*1.5);
    end
end

[M,I] = min(MSEVal);
Minimums(1,:) = M;
Minimums(2,:) = I;
display(Minimums);


prompt = 'What is the best NN number [1 10]?   ';
NN = input(prompt);
prompt = 'What is the best number of nidden nodes?   ';
HN = input(prompt);

for o = 1:NN
    rng(o);
    for i = 1:HN
        ITNum = o + (i/1000);
        display(ITNum);
        % Set up network parameters.
        nin = 7;            % Number of inputs.
        nhidden = i;            % Number of hidden units.
        nout = 3;           % Number of outputs.
        alpha = 0.01;           % Coefficient of weight-decay prior.

        % Create and initialize network weight vector.
        net = mlp(nin, nhidden, nout, 'linear', alpha);

        % Set up vector of options for the optimiser.
        options = zeros(1,18);
        options(1) = -1;            % This provides display of error values.
        options(14) = 1000;     % Number of training cycles.

        % Train using scaled conjugate gradients.
        [net, options, errlog] = netopt(net, options, InputsCVTrain,↙
OutputsCVTrain, 'scg');  %Also use scg
        [eTr, edataTr, epriorTr] = mlperr(net, InputsCVTrain, OutputsCVTrain);
        MSETr2(i,o) = edataTr/(size(InputsCVTrain,1)*1.5);

        yVal = mlpfwd(net, InputsCVVal);
        [eVal, edataVal, epriorVal] = mlperr(net, InputsCVVal, OutputsCVVal);
        MSEVal2(i,o) = edataVal/(size(InputsCVVal,1)*1.5);

        yTest = mlpfwd(net, InputsCVTest);
        [eTest, edataTest, epriorTest] = mlperr(net, InputsCVTest, OutputsCVTest);
        MSETest2(i,o) = edataTest/(size(InputsCVTest,1)*1.5);
    end
end

figure;
plot(yVal(:,3));
```

18/09/17 13:53 D:\MATLA...\NN Tests CrossValidation SE.m   8 of 8

```
hold on;
plot(OutputsCVVal(:,3),'--');
% title({'Training and Prediction values of Crater Wear.'; 'NN number 9 with 13 Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Crater Wear','FontSize',14);
```

APPENDIX XI – FULL TEST DATA NETWORK ALGORITHM

```
clearvars, clc, close all;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%% LOAD DATA %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Import the data from Excel
% Data for first 9 tests
[~, ~, raw] = xlsread('C:\Users\javie\Desktop\Main Tests 2016\Wear Values for↙
Matlab.xlsx','Sheet1','C4:AT15');
stringVectors = string(raw(:,[5,10,15,20,25,30,35,40]));
stringVectors(ismissing(stringVectors)) = '';
raw = raw(:,↙
[1,2,3,4,6,7,8,9,11,12,13,14,16,17,18,19,21,22,23,24,26,27,28,29,31,32,33,34,36,37,38,↙
39,41,42,43,44]);
% Create output variable
data = reshape([raw{:}],size(raw));
% Allocate imported array to column variable names
t = 1;
for i = 1:9
    VBmax(:,i) = data(:,t);
    t = t + 4;
end
t = 4;
for i = 1:9
    VBN(:,i) = data(:,t);
    t = t + 4;
end

% Data for second 8 tests (test 15 is ommited)
[~, ~, raw] = xlsread('C:\Users\javie\Desktop\Main Tests 2016\Wear Values for↙
Matlab.xlsx','Sheet1','C19:AT30');
stringVectors = string(raw(:,[5,10,15,20,25,30,35,40]));
stringVectors(ismissing(stringVectors)) = '';
raw = raw(:,↙
[1,2,3,4,6,7,8,9,11,12,13,14,16,17,18,19,21,22,23,24,26,27,28,29,31,32,33,34,36,37,38,↙
39,41,42,43,44]);
% Create output variable
data = reshape([raw{:}],size(raw));
% Allocate imported array to column variable names
t = 1;
for i = 10:18
    if i < 15
        VBmax(:,i) = data(:,t);
        t = t + 4;
    end
    if i == 15
        t = t + 4;
    end
    if i > 15
        VBmax(:,i-1) = data(:,t);
        t = t + 4;
    end
end
t = 4;
for i = 10:18
    if i < 15
```

```matlab
        VBN(:,i) = data(:,t);
        t = t + 4;
    end
    if i == 15
        t = t + 4;
    end
    if i > 15
        VBN(:,i-1) = data(:,t);
        t = t + 4;
    end
end
VBmax = zscore(VBmax);
VBN = zscore(VBN);
% Clear temporary variables
clearvars data raw stringVectors;

% Load Area, Intensity and Crater Wear data
for i = 1:17
    filename = sprintf('Area3%02d.mat', i);
    Inputs(i).Area = importdata(filename)';
    filename = sprintf('Intensity3%02d.mat', i);
    Inputs(i).Int = importdata(filename)';
    filename = sprintf('MaxAreamm%02d.mat', i);
    Craters(2:12,i) = importdata(filename)';
    Craters(1,i) = 0;
end
Craters = zscore(Craters);
t = 0:11;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%% FIT CURVES & MACHINING PARAMETERS %%%%%%%%%%%%%%%%%%%%%%

%%Fit curves with cubic interpolation
for i = 1:17
    s = size(Inputs(i).Area,1);
    t = 0:(s/11):s;
    t = t';
    Time(i).tt = [0:(s-1)]';
    fitobject = fit(t,VBN(:,i),'linearinterp');   %Use either cubicinterp or linearinterp
    Outputs(i).VBN = feval(fitobject,Time(i).tt);
    fitobject = fit(t,VBmax(:,i),'linearinterp');
    Outputs(i).VBmax = feval(fitobject,Time(i).tt);
    fitobject = fit(t,Craters(:,i),'linearinterp');
    Outputs(i).Crater = feval(fitobject,Time(i).tt);
%     figure; plot(time(i).tt,Craters2(i).Crater,t,Craters(:,i));     %Use only to evaluate fitting
%     figure; plot(time(i).tt,VBmax2(i).VBmax,t,VBmax(:,i));

    % Create normalised machining variables and load into the 'Input'.
    %Cutting Speed variables
    if i== 2 || i==3 || i==8 || i==9 || i==10
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).Cut = Value';
        clear Value;
    end
    if i== 4 || i==6 || i==12 || i==15 || i==16 || i==17
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).Cut = Value';
        clear Value;
    end
    if i== 1 || i==5 || i==7 || i==11 || i==13 || i==14
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).Cut = Value';
        clear Value;
    end

    %Feed variables
    if i== 5 || i==8 || i==9 || i==11 || i==12 || i==17
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).Feed = Value';
        clear Value;
    end
    if i== 1 || i==2 || i==3 || i==6 || i==13 || i==15
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).Feed = Value';
        clear Value;
    end
    if i== 4 || i==7 || i==10 || i==14 || i==16
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).Feed = Value';
        clear Value;
    end

    %Cutting Depth variables
    if i== 2 || i==6 || i==7 || i==8 || i==11 || i==16
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).Depth = Value';
        clear Value;
    end
    if i== 1 || i==9 || i==10 || i==12 || i==14 || i==15
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).Depth = Value';
        clear Value;
```

20/09/17 10:53   G:\Main Tests 2016\Neural Network V1.m   4 of 10

```matlab
    end
    if i== 3 || i==4 || i==5 || i==13 || i==17
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).Depth = Value';
        clear Value;
    end

    %Radial Immersion variables
    if i== 6 || i==8 || i==10 || i==13 || i==14 || i==17
        for k = 1:s
            Value(k) = -1;
        end
        Inputs(i).RadIm = Value';
        clear Value;
    end
    if i== 1 || i==2 || i==4 || i==11 || i==12
        for k = 1:s
            Value(k) = 0;
        end
        Inputs(i).RadIm = Value';
        clear Value;
    end
    if i== 3 || i==5 || i==7 || i==9 || i==15 || i==16
        for k = 1:s
            Value(k) = 1;
        end
        Inputs(i).RadIm = Value';
        clear Value;
    end
end

for i = 1:17
    InputsL(i).Area = Inputs(i).Area+10;
end

% Flaking event detection and input creation
per = 0.97;
for i = 1:17
    count = 0;
    s = size(InputsL(i).Area,1);
    SparkEvent = zeros(s,1);
    for k = 6:(s-4)
        if (InputsL(i).Area(k-5)*per)&&(InputsL(i).Area(k-4)*per)&&(InputsL(i).Area(k-3)*per)&&...
                (InputsL(i).Area(k-2)*per)&&(InputsL(i).Area(k-1)*per) > ...
                (InputsL(i).Area(k))&&(InputsL(i).Area(k+1))&&(InputsL(i).Area(k+2))&&...
                (InputsL(i).Area(k+3))&&InputsL(i).Area(k+4)
            SparkEvent(k:s,1) = (count + 1)/100;
            count = count + 1;
        end
    end
    Inputs(i).SE = SparkEvent;
```

```
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%% GENERAL MATRICES FOR TRAINING, VALIDATION AND TESTING %%%%%%%%%%%

% Create matrices with columns correpsonding to the 6 inputs in the first,
% and 3 outputs in the second; and columns corresponding to the
% time stamps of all the 17 tests concatenated
% JUST TRAINING TESTS  = 11 (Validation = 3, Testing = 3)
% Taken from L9 array A (number 15 corresponds to 16 in the table array)
% and the 3, 9, 14 and 17 (18 in the table array) tests of L9 array B, section↙
3.3.3.1
TotalSizes = 0;
for i = [ 2 3 4 5 7 8 9 10 12 13 14 15 17 ]
    s = size(Inputs(i).Area,1);
    TotalSizes = TotalSizes + s;
end
% Build training inputs matrix
InputsMatTrain = zeros(TotalSizes,6);
Inc = 0;
for i = [ 2 3 4 5 7 8 9 10 12 13 14 15 17 ]
    k = 1;
    s = size(Inputs(i).Area,1);
    InputsMatTrain(Inc + 1:Inc + s,k) = Inputs(i).Area;
    k = k + 1;
    s = size(Inputs(i).Int,1);
    InputsMatTrain(Inc + 1:Inc + s,k) = Inputs(i).Int;
    k = k + 1;
    s = size(Inputs(i).Cut,1);
    InputsMatTrain(Inc + 1:Inc + s,k) = Inputs(i).Cut;
    k = k + 1;
    s = size(Inputs(i).Feed,1);
    InputsMatTrain(Inc + 1:Inc + s,k) = Inputs(i).Feed;
    k = k + 1;
    s = size(Inputs(i).Depth,1);
    InputsMatTrain(Inc + 1:Inc + s,k) = Inputs(i).Depth;
    k = k + 1;
    s = size(Inputs(i).RadIm,1);
    InputsMatTrain(Inc + 1:Inc + s,k) = Inputs(i).RadIm;
    k = k + 1;
    s = size(Inputs(i).SE,1);
    InputsMatTrain(Inc + 1:Inc + s,k) = Inputs(i).SE;
    k = k + 1;
    Inc = Inc + s;
end
% Build training outputs matrix
OutputsMatTrain = zeros(TotalSizes,3);
Inc = 0;
for i = [ 2 3 4 5 7 8 9 10 12 13 14 15 17 ]
    k = 1;
    s = size(Outputs(i).VBN,1);
    OutputsMatTrain(Inc + 1:Inc + s,k) = Outputs(i).VBN;
    k = k + 1;
    s = size(Outputs(i).VBmax,1);
    OutputsMatTrain(Inc + 1:Inc + s,k) = Outputs(i).VBmax;
    k = k + 1;
    s = size(Outputs(i).Crater,1);
    OutputsMatTrain(Inc + 1:Inc + s,k) = Outputs(i).Crater;
    k = k + 1;
    Inc = Inc + s;
end

% Create matrices for Validation
% Taken from L9 array B, tests 11 and 16(17 in the table array)
% in section 3.3.3.1
TotalSizes = 0;
for i = [ 11 16 ]
    s = size(Inputs(i).Area,1);
    TotalSizes = TotalSizes + s;
end
% Build validation inputs matrix
InputsMatVal = zeros(TotalSizes,6);
Inc = 0;
for i = [ 11 16 ]
    k = 1;
    s = size(Inputs(i).Area,1);
    InputsMatVal(Inc + 1:Inc + s,k) = Inputs(i).Area;
    k = k + 1;
    s = size(Inputs(i).Int,1);
    InputsMatVal(Inc + 1:Inc + s,k) = Inputs(i).Int;
    k = k + 1;
    s = size(Inputs(i).Cut,1);
    InputsMatVal(Inc + 1:Inc + s,k) = Inputs(i).Cut;
    k = k + 1;
    s = size(Inputs(i).Feed,1);
    InputsMatVal(Inc + 1:Inc + s,k) = Inputs(i).Feed;
    k = k + 1;
    s = size(Inputs(i).Depth,1);
    InputsMatVal(Inc + 1:Inc + s,k) = Inputs(i).Depth;
    k = k + 1;
    s = size(Inputs(i).RadIm,1);
    InputsMatVal(Inc + 1:Inc + s,k) = Inputs(i).RadIm;
    k = k + 1;
    s = size(Inputs(i).SE,1);
    InputsMatVal(Inc + 1:Inc + s,k) = Inputs(i).SE;
    k = k + 1;
    Inc = Inc + s;
end
% Build validation outputs matrix
OutputsMatVal = zeros(TotalSizes,3);
Inc = 0;
for i = [ 11 16 ]
    k = 1;
    s = size(Outputs(i).VBN,1);
    OutputsMatVal(Inc + 1:Inc + s,k) = Outputs(i).VBN;
    k = k + 1;
    s = size(Outputs(i).VBmax,1);
    OutputsMatVal(Inc + 1:Inc + s,k) = Outputs(i).VBmax;
    k = k + 1;
    s = size(Outputs(i).Crater,1);
    OutputsMatVal(Inc + 1:Inc + s,k) = Outputs(i).Crater;
    k = k + 1;
    Inc = Inc + s;
end

% Create matrices for Testing
% Taken from L9 array B, tests 1 and 6
% in section 3.3.3.1
TotalSizes = 0;
for i = [ 1 6 ]
    s = size(Inputs(i).Area,1);
    TotalSizes = TotalSizes + s;
end
% Build testing inputs matrix
InputsMatTest = zeros(TotalSizes,6);
Inc = 0;
for i = [ 1 6 ]
    k = 1;
    s = size(Inputs(i).Area,1);
    InputsMatTest(Inc + 1:Inc + s,k) = Inputs(i).Area;
    k = k + 1;
    s = size(Inputs(i).Int,1);
    InputsMatTest(Inc + 1:Inc + s,k) = Inputs(i).Int;
    k = k + 1;
    s = size(Inputs(i).Cut,1);
    InputsMatTest(Inc + 1:Inc + s,k) = Inputs(i).Cut;
    k = k + 1;
    s = size(Inputs(i).Feed,1);
    InputsMatTest(Inc + 1:Inc + s,k) = Inputs(i).Feed;
    k = k + 1;
    s = size(Inputs(i).Depth,1);
    InputsMatTest(Inc + 1:Inc + s,k) = Inputs(i).Depth;
    k = k + 1;
    s = size(Inputs(i).RadIm,1);
    InputsMatTest(Inc + 1:Inc + s,k) = Inputs(i).RadIm;
    k = k + 1;
    s = size(Inputs(i).SE,1);
    InputsMatTest(Inc + 1:Inc + s,k) = Inputs(i).SE;
    k = k + 1;
    Inc = Inc + s;
end
% Build testing outputs matrix
OutputsMatTest = zeros(TotalSizes,3);
Inc = 0;
for i = [ 1 6 ]
    k = 1;
    s = size(Outputs(i).VBN,1);
    OutputsMatTest(Inc + 1:Inc + s,k) = Outputs(i).VBN;
    k = k + 1;
    s = size(Outputs(i).VBmax,1);
    OutputsMatTest(Inc + 1:Inc + s,k) = Outputs(i).VBmax;
    k = k + 1;
    s = size(Outputs(i).Crater,1);
    OutputsMatTest(Inc + 1:Inc + s,k) = Outputs(i).Crater;
    k = k + 1;
```

20/09/17 10:53 G:\Main Tests 2016\Neural Network V1.m 8 of 10

```
    Inc = Inc + s;
end

BestNodes = ((size(OutputsMatTrain,1)/8)-size(OutputsMatTrain,2))/(size↙
(InputsMatTrain,2)+size(OutputsMatTrain,2)+1);
MinNodes = ((size(OutputsMatTrain,1)/3)-size(OutputsMatTrain,2))/(size↙
(InputsMatTrain,2)+size(OutputsMatTrain,2)+1);
MidNodes = ((size(OutputsMatTrain,1)/2)-size(OutputsMatTrain,2))/(size↙
(InputsMatTrain,2)+size(OutputsMatTrain,2)+1);
MaxNodes = ((size(OutputsMatTrain,1))-size(OutputsMatTrain,2))/(size↙
(InputsMatTrain,2)+size(OutputsMatTrain,2)+1);
display(BestNodes);
display(MinNodes);
display(MidNodes);
display(MaxNodes);
prompt = 'What is the maximum number of hidden nodes for evaluation?   ';
MaxHN = input(prompt);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% NEURAL NETWORKS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Cross validation to get number of hidden units.
for o = 1:10
    rng(o);
    for i = 1:MaxHN
        ITNum = o + (i/1000);
        display(ITNum);
        % Set up network parameters.
        nin = 7;            % Number of inputs.
        nhidden = i;            % Number of hidden units.
        nout = 3;           % Number of outputs.
        alpha = 0.01;           % Coefficient of weight-decay prior.

        % Create and initialize network weight vector.
        net = mlp(nin, nhidden, nout, 'linear', alpha);

        % Set up vector of options for the optimiser.
        options = zeros(1,18);
        options(1) = -1;                % This provides display of error values.
        options(14) = 1000;     % Number of training cycles.

        % Train using scaled conjugate gradients.
        [net, options, errlog] = netopt(net, options, InputsMatTrain,↙
OutputsMatTrain, 'scg');  %Also use scg
        yTr = mlpfwd(net, InputsMatTrain);
        [eTr, edataTr, epriorTr] = mlperr(net, InputsMatTrain, OutputsMatTrain);
        MSETr(i,o) = edataTr/(size(InputsMatTrain,1)*1.5);

        yVal = mlpfwd(net, InputsMatVal);
        [eVal, edataVal, epriorVal] = mlperr(net, InputsMatVal, OutputsMatVal);
        MSEVal(i,o) = edataVal/(size(InputsMatVal,1)*1.5);

        yTest = mlpfwd(net, InputsMatTest);
        [eTest, edataTest, epriorTest] = mlperr(net, InputsMatTest,↙
OutputsMatTest);
```

```
        MSETest(i,o) = edataTest/(size(InputsMatTest,1)*1.5);
    end
end

prompt = 'What is the best NN number [1 10]?   ';
NN = input(prompt);
prompt = 'What is the best number of nidden nodes?   ';
HN = input(prompt);

for o = 1:NN
    rng(o);
    for i = 1:HN
        ITNum = o + (i/1000);
        display(ITNum);
        % Set up network parameters.
        nin = 7;            % Number of inputs.
        nhidden = i;            % Number of hidden units.
        nout = 3;           % Number of outputs.
        alpha = 0.01;           % Coefficient of weight-decay prior.

        % Create and initialize network weight vector.
        net = mlp(nin, nhidden, nout, 'linear', alpha);

        % Set up vector of options for the optimiser.
        options = zeros(1,18);
        options(1) = -1;            % This provides display of error values.
        options(14) = 1000;     % Number of training cycles.

        % Train using scaled conjugate gradients.
        [net, options, errlog] = netopt(net, options, InputsMatTrain,↙
OutputsMatTrain, 'scg');  %Also use scg
        yTr = mlpfwd(net, InputsMatTrain);
        [eTr, edataTr, epriorTr] = mlperr(net, InputsMatTrain, OutputsMatTrain);
        MSETr2(i,o) = edataTr/(size(InputsMatTrain,1)*1.5);

        yVal = mlpfwd(net, InputsMatVal);
        [eVal, edataVal, epriorVal] = mlperr(net, InputsMatVal, OutputsMatVal);
        MSEVal2(i,o) = edataVal/(size(InputsMatVal,1)*1.5);

        yTest = mlpfwd(net, InputsMatTest);
        [eTest, edataTest, epriorTest] = mlperr(net, InputsMatTest,↙
OutputsMatTest);
        MSETest2(i,o) = edataTest/(size(InputsMatTest,1)*1.5);
    end
end

figure;
plot(yVal(:,1));
hold on;
plot(OutputsMatVal(:,1),'--');
% title({'Validation and Prediction values of Notch Wear.'; 'NN number 9 with 13↙
Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Notch Wear','FontSize',14);
```

20/09/17 10:53   G:\Main Tests 2016\Neural Network V1.m   10 of 10

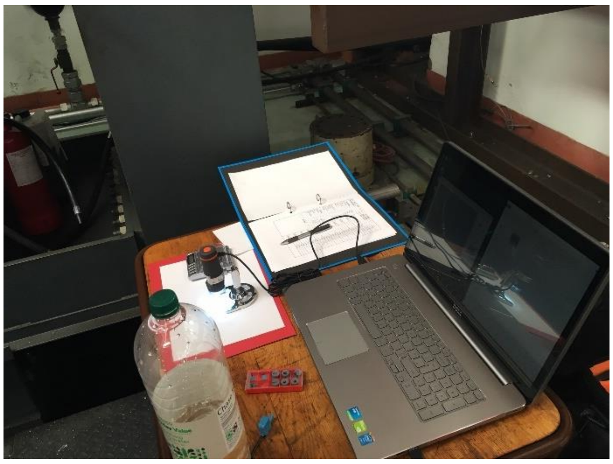
```
figure;
plot(yVal(:,2));
hold on;
plot(OutputsMatVal(:,2),'--');
% title({'Validation and Prediction values of Flank Wear.'; 'NN number 9 with 13 Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Flank Wear','FontSize',14);
figure;
plot(yVal(:,3));
hold on;
plot(OutputsMatVal(:,3),'--');
% title({'Validation and Prediction values of Crater Wear.'; 'NN number 9 with 13 Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Crater Wear','FontSize',14);

figure;
plot(yTest(:,1));
hold on;
plot(OutputsMatTest(:,1),'--');
% title({'Validation and Prediction values of Notch Wear.'; 'NN number 9 with 13 Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Notch Wear','FontSize',14);
figure;
plot(yTest(:,2));
hold on;
plot(OutputsMatTest(:,2),'--');
% title({'Validation and Prediction values of Flank Wear.'; 'NN number 9 with 13 Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Flank Wear','FontSize',14);
figure;
plot(yTest(:,3));
hold on;
plot(OutputsMatTest(:,3),'--');
% title({'Validation and Prediction values of Crater Wear.'; 'NN number 9 with 13 Hidden Nodes.'},'FontSize',14);
legend('Network Prediction','Original Values','Location','northwest')
xlabel('Data Points','FontSize',14);
ylabel('Crater Wear','FontSize',14);
```

APPENDIX XII – PHOTOS OF APARATUS SET-UP IN THIRD SET OF EXPERIMENTS

- Camera set-up.

- Inconel 718 workpiece mounted and prepared with square shape for testing.

- In-situ wear assessment and microscope imaging.



- Cutting tool.