INSTITUTO TECNOLÓGICO DE AERONÁUTICA

Adriana Nunes Chaves Lima

COMPUTATIONAL METHOD FOR TEMPERATURES AND HEAT FLOWS ANALYSIS OF ORTHOGONAL CUTTING 1045 STEEL BY THERMAL IMAGING

Final Paper
2017

Course of Mechanical Engineering

CDU

Adriana Nunes Chaves Lima

COMPUTATIONAL METHOD FOR TEMPERATURES AND HEAT FLOWS ANALYSIS OF ORTHOGONAL CUTTING 1045 STEEL BY THERMAL IMAGING

Advisor

Prof. Dr. Anderson Vicente Borille (ITA)

Co-advisor

Dipl. Wirt. Ing. Thorsten Augspurger (WZL)

MECHANICAL ENGINEERING

São José dos Campos
Instituto Tecnológico de Aeronáutica

2017

Cataloging-in Publication Data
Documentation and Information Division

Nunes Chaves Lima, Adriana
Computational method for temperatures and heat flows analysis of orthogonal cutting 1045 steel by thermal imaging / Adriana Nunes Chaves Lima.
São José dos Campos, 2017.
64f.

Final paper (Undergraduation study) – Course of Mechanical Engineering– Instituto Tecnológico de Aeronáutica, 2017. Advisor: Prof. Dr. Anderson Vicente Borille. Co-advisor: Dipl. Wirt. Ing. Thorsten Augspurger.

1. Usinagem. 2. Corte Ortogonal. 3. Programas. 4. Análise Térmica. 5. Estado Trasiente. I. Instituto Tecnológico de Aeronáutica. II. Title.

BIBLIOGRAPHIC REFERENCE

NUNES CHAVES LIMA, Adriana. **Computational method for temperatures and heat flows analysis of orthogonal cutting 1045 steel by thermal imaging**. 2017. 64f. Final paper (Undergraduation study) – Instituto Tecnológico de Aeronáutica, São José dos Campos.

CESSION OF RIGHTS

Adriana Nunes Chaves Lima
Av. Dr. Eduardo Cury, 350, apt. 257
12.242-001 – São José dos Campos–SP

COMPUTATIONAL METHOD FOR TEMPERATURES AND HEAT FLOWS ANALYSIS OF ORTHOGONAL CUTTING 1045 STEEL BY THERMAL IMAGING

This publication was accepted like Final Work of Undergraduation Study

Adriana Nunes Chaves Lima
Author

Anderson Vicente Borille (ITA)
Advisor

Thorsten Augspurger (WZL)
Co-advisor

Prof. Dr. Jesuíno Takashi Tomita
Course Coordinator of Mechanical Engineering

São José dos Campos: November 20, 2017.

I dedicate this work to my family, which have always supported me in my decisions and are the most happy ones with this academic achievement.

Acknowledgments

"The man that makes no mistakes does not usually do anything."

— Bishop W.C.Magee

Resumo

Métodos de inspeção e monitoramento têm sido utilizados cada vez mais para garantir a qualidade de processos. No campo da usinagem existem muitos parâmetros importantes para assegurar que o processo forneça os resultados estimados. O acabamento superficial de uma peça usinada e a vida útil de uma ferramenta, por exemplo, sofrem influência direta da energia térmica gerada nas zonas de calor. Devido a isso, existem muitos métodos teóricos para a modelagem de temperatura distribuída pela zona de corte, mas ainda faltam ferramentas que possam permitir a validação prática de tais métodos. Embora ainda existam desafios no uso adequado da termografia, essa tecnologia faz possível o desenvolvimento de métodos computacionais para o processamento de imagens térmicas e, consequentemente a posterior análise de fluxos de calor e partições dessa energia. Este trabalho apresenta um método computacional desenvolvido em MATLAB, com o suporte da toolbox de processamento de imagens, para análise de imagens térmicas, fornecendo resultados de campos de temperatura, energias internas, fluxos de calor e outras variáveis de interesse que possam ser utilizadas no monitoramento da usinagem e no estudos de melhores parâmetros de corte.

Abstract

Methods for inspection and monitoring have been used more and more to ensure the quality of processes. In the machining field there are many important process parameters to assure that the expected results are achieved. The surface quality of a workpiece and the tool life, for example, are directly influenced by the thermal energy generated in the heat zones. Due to it, there are lots of theoretical methods for temperature modeling along the cutting zone, but there is still a lack of tools able to allow for practical validation of these methods. Although many challenges still prevail for the adequate use of thermography, this technology enables the development of computational methods for processing of thermal images and, consequently, the heat flow and heat partition analysis. This paper comes to present a computational method developed on MATLAB with the support of image processing toolbox. It performs thermal image analysis, providing results regarding temperature fields, inner energies, heat flows and other variables of interest that can be used on machining monitoring and future studies to improve cutting parameters.

List of Figures

List of Tables

List of Abbreviations and Acronyms

MATLAB	Numerical computation software from MathWorks
GUI	graphic user interface
GUIDE	graphic user interface development environment
AISI	American iron and steel institute
WZL	Werkzeugmaschinenlabor (Laboratory of Machine Tools)
FOV	Field of view
fps	Frames per second
HSM	High speed machining
ROI	Region of interest

List of Symbols

F_c	Cutting force on the power direction $[N]$
F_p	Passive force $[N]$
v_c	Cutting velocity $[m/min]$
v_{chip}	Exit velocity of chip $[m/min]$
P	Total power developed along cutting process $[W]$
w	Width of tool $[mm]$
a_p	Depth of cut $[\mu m]$
t_c	Chip thickness $[\mu m]$
T_e	Environment temperature $[^oC]$
k	Heat conductivity of tool material $[W/mK]$
c_p^T	Heat capacity of tool $[J/cm^3K]$
c_p^W	Heat capacity of workpiece $[J/kgK]$
α	Rake angle $[^o]$
γ	Clearance angle $[^o]$
ϕ	Shear angle $[^o]$
r_β	Cutting edge radius $[\mu m]$
ϵ	Emissivity
L	Length of chosen isotherm [pixel]
$\frac{dT}{dz}$	Variation of temperature along normal of chosen isotherm $[^oC/pixel]$
$\dot{Q}_T$	Heat flow through tool $[W]$
$\dot{Q}_C^{out}$	Energy carried away by chip $[W]$
$\dot{Q}_C^{in}$	Energy carried in by chip $[W]$
$\dot{Q}_W$	Heat flow through workpiece $[W]$
$\dot{Q}_{inside}$	Total energy into the control volume$[W]$
$\dot{Q}_{outside}$	Total energy out to the control volume$[W]$
$\dot{Q}_{shear}$	Total energy generated in the primary shear zone$[W]$
T_C^{out}	Temperature of chip along line of end of contact $[^oC]$
p_T	Partition of heat that goes to tool
p_C	Partition of heat that goes to chip
p_W	Partition of heat that goes to workpiece

ρ	Parametric distance from reference axis to line in hough transform
θ	Angular coefficient of lines in the reference axis in hough transform

Contents

1 Introduction

There are different ways to modify raw material, such as additive and subtractive methods (SHAW; COOKSON, 2005). The additive processes occur when separate materials are put together, like in 3D printing or welding. On the other hand, the subtractive precesses remove unnecessary material, which happens for machining processes such as turning, milling and, as discussed in this paper, orthogonal cutting.

In many machining cases, orthogonal cutting may be considered a good approximation to perform on the major cutting edge, that is why it has been extensively studied (SHAW; COOKSON, 2005). For instance, planing and facing processes are some examples in which orthogonal cutting conditions can be observed.

The cutting zone is composed basically by chip, tool and workpiece. Many parameters are responsible for a good performance and final result of machining processes, as well as for good surface finishing of workpieces. Depth of cut, cutting velocity, cutting material are some of these parameters. It is fundamental to use the right parameters for each type of cutting process, otherwise it can affect negatively the expected result and the process itself.

Also, it is known that high temperatures and thermal behavior during cutting processes have a strong influence on tool life, surface finish, metallurgical structure of the workpiece, machinability, tool wear and thermal deformation of the tool, which is the largest source of errors in machining processes. However, the knowledge concerning machining of metals is not yet fully understood. Some questions about the location and shape of heat sources and the effects of the combination of deformation and temperature distribution still prevail.

Many studies have been conducted in order to measure temperature fields to a better understanding of the thermal behavior in the cutting zone. There are several ways of obtaining temperature measurements of the cutting zone. A critical review is made in (KOMANDURI; HOU, 2000), (KOMANDURI; HOU, 2001), (ABUKHSHIM *et al.*, 2006).

Thermocouple method uses two dissimilar metals that are put together, making two junctions. When the junctions have different temperatures, an electromotive force is generated, which its value depends on the material used in the thermocouples and the temperatures in each junction. This method has advantages such as low cost, simplicity in operation and in construction. On the other hand, for embedded thermocouples it is necessary to make fine holes in the tool structure, which interfere in heat and temperature measurements (KOMANDURI; HOU, 2001).

Another way to measure temperature during machining is by the infrared photographic technique developed by Boothroyd (1961). It is able to measure temperature fields on the shear zone and tool-chip interface. The method uses an infrared sensitive photographic plate to capture information from the cutting zone and then measures the density of this plate with a microdensitometer. However, this technique does not allow for a fast inspection, because the acquisition rate is low for a HSM, which demands a high fps to get enough thermal images in order to analyze the entire cutting process.

There are also thermal paintings capable of change their color according to the temperature. It is very simple to apply and cheap. But this is a method to be used on systems with controlled heat conditions (KOMANDURI; HOU, 2001).

The use of radiation techniques is interesting for cutting processes with high velocities. It has a fast response in getting temperature distribution all over the cutting surface and does not require any kind of contact with the object of interest, which makes it the most suitable method for temperature measurement in HSM (ABUKHSHIM *et al.*, 2006). Due to the high velocities used in the experiments discussed in this paper, the infrared camera will be the tool used for temperature measurement in the orthogonal cutting experiments.

1.1 Objective

The aim of this paper is to develop a computational method to analyze thermal images generated during orthogonal cutting of AISI 1045 steel, which focus on the transient state due to the short cutting time. The data that will be analyzed are the temperature distribution along the cutting tool, and the heat flows through tool, chip and workpiece. The method comes to provide a fast implementation tool to be used in validation of future studies on heat generation in cutting zones.

1.2 Structure

This work is divided into 6 Chapters, including this **Introduction**, plus one Appendix.

The second chapter, **Bibliographic Review**, describes the existing technology which is relevant for the scope of this paper.

The third chapter, **Materials and Methods**, describes the materials and methods that guided the experiments.

The fourth, **Results**, presents the results and discussions about code implementation and model validation.

The fifth and final chapter, **Conclusions**, sums up what was accomplished in this work and suggests how it may be expanded for new processes.

The Appendix **Source Code** contains all the code written for the program.

2 Bibliographic Review

2.1 Thermal review

2.1.1 Heat zones in machining

In machining there are 3 main regions of interest from where comes the heat produced during the cutting process (SHAW; COOKSON, 2005). The first area represented on figure 2.1 is called primary shear zone and it is located along the shear plane, which is the boundary between undeformed workpiece and chip. The second area is the contact plane between tool and chip, also known as secondary shear zone or friction zone. As for the third one, it is related to the wear caused due to the friction between tool and finished workpiece surface. It is called wear zone or tertiary zone.

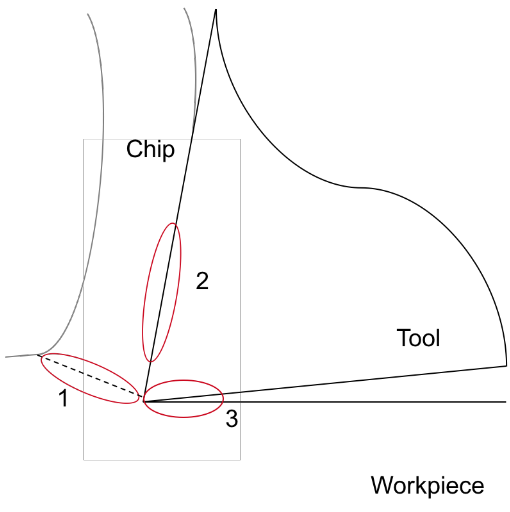

FIGURE 2.1 – Regions of interest during tool-chip interaction

All the heat generated in these three zones is removed from the system by means of tool, chip and workpiece. Since the cutting velocities used in this study are high, the cutting process is classified as high speed machining. The process is more adiabatic as the cutting velocity raises because of the short process time prevents the heat of being dissipated from the heat source directly to the environment.

It is important to highlight that the heat generation on primary and secondary shear zones is highly influenced by the cutting conditions, while for the tertiary zone is mainly dependent on tool flank wear (ABUKHSHIM *et al.*, 2006). In this study, each cutting experiment was performed with sharp tools, so that the wear zone had a minor influence on total heat generation (SHAW; COOKSON, 2005).

2.1.2 Fundamentals of heat transfer

Heat transfer occurs in three basic ways: Conduction, convection and radiation. Each situation can present one or more of these modes happening at the same time (POOLE; SARVAR, 1989). Regarding conduction, this is a mechanism in which heat is transferred from a region with high temperatures to another region with lower temperatures in a material. The general equation for heat conduction in three dimensions is given by:

$$k(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2}) + q = \rho c_p \frac{\partial T}{\partial t} \tag{2.1}$$

Where q is the heat generated per volume, k is the heat conductivity, T is temperature, t is time, ρ is the density of the material, c_p is the specific heat capacity and x, y and z are the directions of heat propagation.

Convection is the way of heat propagation between bodies and fluids and within fluids. It happens because of density difference caused by the temperature difference. The equation that rules this mode is:

$$q = h_c A(T_f - Ts) \tag{2.2}$$

Where h_c is the convective heat transfer coefficient, A is the area of the body in contact with the fluid and T_f and T_s are the temperatures of the fluid and surface, respectively.

For the third mode, the presence of a transport medium is not necessary. Radiation makes it possible for heat transfer in vacuum and any body above absolute zero emits electromagnetic energy causing heat propagation. Given two bodies with absolute temperatures T_1 and T_2, the heat propagation is:

$$q = \epsilon \sigma_B A(T_1^4 - T_2^4) \tag{2.3}$$

σ_B is the Stefan-Boltzmann constant, ϵ is the emissivity and A is the enclosed area of the body.

Besides radiation, which comes to be the prevailing mode in infrared thermography, which will be discussed on the following subsection 2.1.3, the focus of this paper is in the conduction transfer present in the cutting zone.

2.1.3 Infrared thermography operation

Infrared termography is a non-contact way of measuring infrared electromagnetic energy. The human eye cannot detect the range of infrared radiation. However, there are infrared cameras which are able to detect this energy and process the radiation into visual information (figure 2.2).

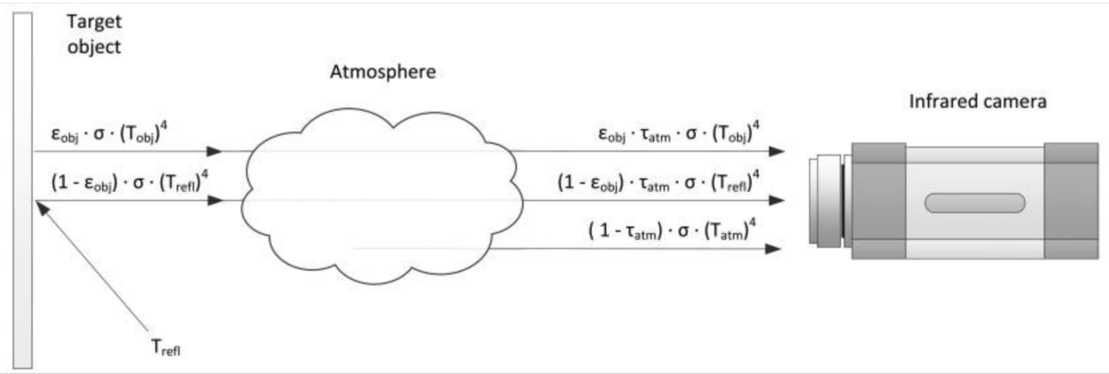

FIGURE 2.2 – Radiation received by infrared camera (USAMENTIAGA *et al.*, 2014)

It makes it possible for all thermal energy produced during a cutting process to be received by the infrared camera and to be later synthesize into a temperature matrix. Since every body is able to emit infrared radiation when its temperature is above absolute zero, it is possible to observe contours of different bodies due to their temperature distribution. For this reason, thermography is a very important technology in military use, because it allows objects to be seen even without proper illumination or in total lack of light situations.

Thermography is able to work in two different ways: passive and active. The passive variety occurs when the subject has its temperature different from the environment (often higher). On the other hand, active thermography needs an external heat source to induce a reasonable contrast between the object and the background (MALDAGUE, 2000).

As it can be observed on Figure 2.2, there are external sources of infrared radiation that can interfere in the target's temperature measurement. To correct this situation, the IR camera has an internal process called compensation (USAMENTIAGA *et al.*, 2014).

The total energy received (W_{tot}) is composed by the sum of three parts: the emission from the main object (E_{obj}), the emission of the vicinity reflected by the object (E_{refl}) and the emission of the atmosphere (E_{atm}) as shown on figure 2.2. Then it is possible to extract the real temperature of the target object (USAMENTIAGA *et al.*, 2014).

2.2 Mechanical review

2.2.1 Mechanics of orthogonal cutting

In this section it will be shown innumerous relations among forces, stresses and dimensions, for example. For this purpose it is important to discuss geometrical correlations in the composite cutting force circle (figure 2.3).

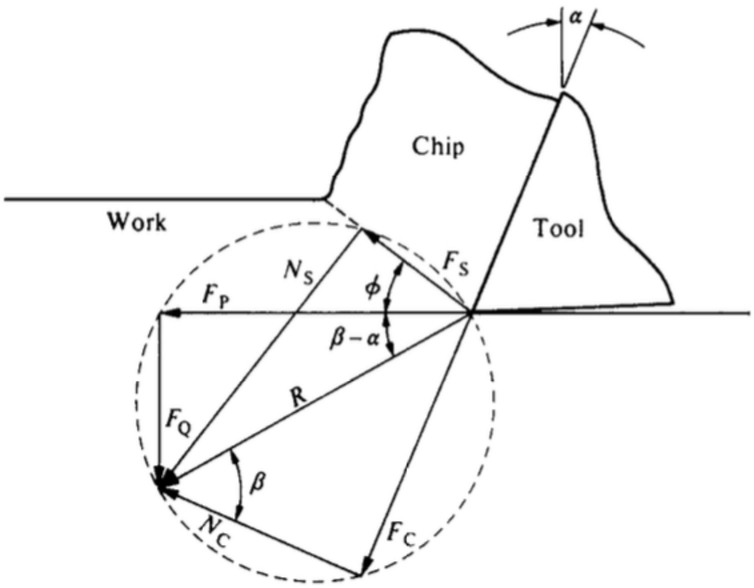

FIGURE 2.3 – Cutting forces (SHAW; COOKSON, 2005)

From the figure 2.3 it can be stated about forces on the primary shear zone reference F_S and N_S:

$$F_S = F_P \cos\phi - F_Q \sin\phi \tag{2.4}$$

$$N_S = F_Q \cos\phi + F_P \sin\phi \tag{2.5}$$

Also, for the forces on the chip flow direction reference:

$$F_C = F_P \sin\alpha + F_Q \cos\alpha \tag{2.6}$$

$$N_C = F_P \cos\alpha - F_Q \sin\alpha \tag{2.7}$$

These equations provide all auxiliary forces related to the known passive force F_Q

and the force on the cutting direction F_P. Now the variables of interest can be easily calculated, such as the friction coefficient:

$$\mu = \frac{F_C}{N_C} = \frac{F_Q + F_P \tan\alpha}{F_P - F_Q \tan\alpha} \tag{2.8}$$

The equations concerning stresses are:

$$A_S = \frac{wa_p}{\sin\phi} \tag{2.9}$$

$$\tau = \frac{F_S}{A_S} = \frac{(F_P \cos\phi - F_Q \sin\phi)\sin\phi}{wa_p} \tag{2.10}$$

$$\sigma = \frac{N_S}{A_S} = \frac{(F_P \sin\phi + F_Q \cos\phi)\sin\phi}{wa_p} \tag{2.11}$$

Where A_S is the area of the shear plane, τ is the shear stress and σ is the normal stress.

Another important parameter is the cutting ratio r, which can provide an important relation between the main cutting velocity and the chip outlet velocity. It has been found experimentally that there is no change in density of metal during the cutting process and also that $w/a_p \geq 5$ makes the width of the chip the same than that of the workpiece. Thus, the equations are:

$$a_p w l = a_{pc} w_c l_c \tag{2.12}$$

Where a_p, w and l are the depth of cut, width of cut and length of cut, respectively. Then, the cutting ratio is defined by:

$$r = \frac{a_p}{a_{pc}} = \frac{l_c}{l} \tag{2.13}$$

With the cutting ratio, it is now possible to correlate cutting velocity v and chip outlet velocity v_c by means of the following equation:

$$v_c = rv \tag{2.14}$$

2.3 State of the Art

2.3.1 Infrared Termography

For the use of infrared thermography it can be found many studies for inspection application. The infrared camera makes it possible to work with thermal information in entire areas covered by the field of view, different from thermocouples which are able to measure punctual temperatures, for example.

Lee *et al.* (2011) shows a study on integrity of resistance spot welding by means of infrared thermography. Two external heat sources were set in order to raise the temperature of the spot. The results have shown a promising method of inspection when it comes to diameter measurement of the nugget. While measurements made with naked eye provide an error about 20%, the thermography provides only 8%.

Also Lebar *et al.* (2010) developed a method that allows online thermal measurement of abrasive water jet cutting. The method makes it possible to extract features from thermal image and to correlate them with texture analysis of the workpiece afterwards. This is important to evaluate the cutting process performance.

Abukhshim *et al.* (2006) summarized general methods for temperature measurement of the shear zone and tool-chip interface, but it points thermography methods to be the most suitable for high speed maching. There is no contact with the heat souces, preventing any external influences, in comparison with other methods, and the temperatures can be processed faster.

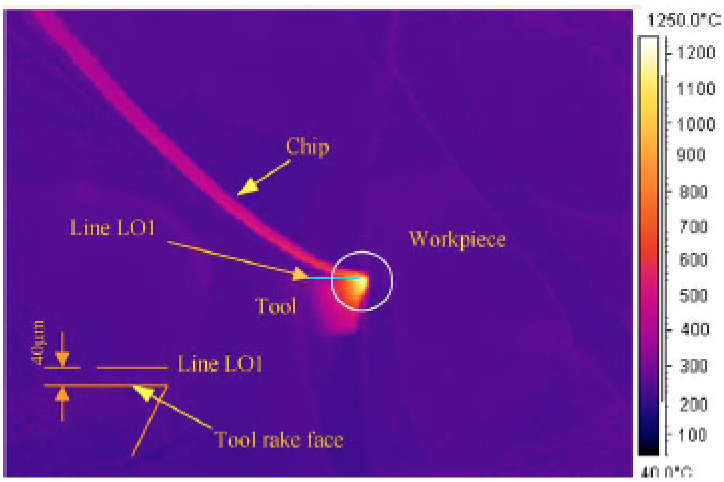

FIGURE 2.4 – Infrared photography of a cutting process (ABUKHSHIM *et al.*, 2006)

For the case under study, high speed thermography has its positive and negative points. On the positive side, it may be mentioned:

- Fast inspection rate (reasonable number of images of high speed cutting)
- Contactless (no interference during the cutting process)
- Easy interpretation of the results (indexed image with temperatures in each pixel)

But it is also important to mention the difficulties that in this method still prevail:

- Only a limited thickness can be measured (under the main surface)
- Determiningl a suitable emissivity is a chalenge (it changes with temperature variation)

2.3.2 Image Processing

Machine vision systems have often been approached with the current fast technology development and intelligent systems. They are used for the most diverse segments, such as the military and medical areas. Image processing has quickly gaining ground. For instance, this is essential when comes to finding a pattern or extract a specific feature in an image.

Colored or gray scaled images can be treated as matrices with dimensions given by their pixel resolution. Each pixel corresponds to a cell inside this matrix and each cell contains a relevant information, which could be a level in grayscale, a coordinate or a temperature as in this paper. Since they are matrices, they can be easily manipulated by means of mathematical operations and consequently processed to highlight one specific property or more.

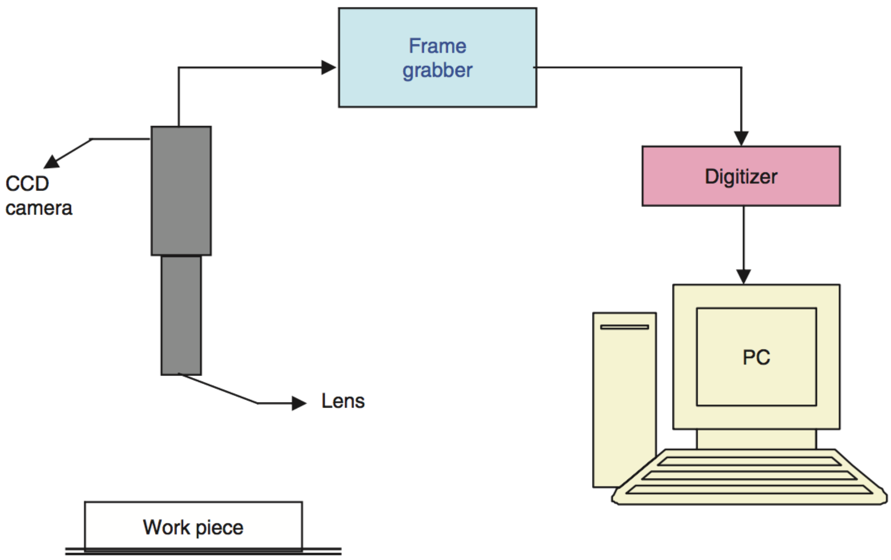

FIGURE 2.5 – Diagram of a machine vision system (SARMA *et al.*, 2009)

There are many applications for image processing in the machining industry. Sarma *et al.* (2009) developed a method for roughness determination (R_a), correlating gray scaled images with surface finish of glass fiber reinforced polymer (GFRP). After GFRP machining, images of the workpiece were taken by means of charge couple device camera and then processed (figure 2.5), obtaining a significant correlation between the predicted and real roughness.

Jeon e Kim (1988) and Kurada e Bradley (1997) also developed an image processing method to monitor flank wear of cutting tools *in situ*. Images in grayscale were taken and consequently processed for boundaries extraction, which indicates wear areas on tool tip surroundings.

Also, Khalifa *et al.* (2006) presented a method for chatter identification in turning processes, which is a significant challenge when comes to automatic machining processes. The vision system compares surface finish of workpieces machined under chatter and chatter-free conditions by means of roughness parameter. The method is also based on the behavior and distribution of gray levels in images of the workpiece.

These are a few examples of what image processing can do for machining industry. There are uncountable other ways in which it can be applied to improve processes and quality of final products. The fast development of computer hardware makes the processing time of images continuously shorter, allowing vision systems to be incorporated in online monitoring and providing real time feedback.

3 Materials and Methods

3.1 Experimental Setup and Materials

The experiments were carried out on Werkzeugmaschinenlabor (WZL) shop floor, located in Aachen in Germany, acquiring thermal images by means of a high speed infrared camera FLIR SC7600 (with frame rate of 328 fps and a resolution of 640 x 512 pixels), equipped with a macro lens 1:1 and FOV 9.6 x 7.7 mm. The test bench is set up that the tool stays in a fixed position in relation to the camera, keeping the relative distance between tool and camera constant. The scale factor provided by this setting was 15 μm/pixel. It allows for the metric conversion for future post processing of images.

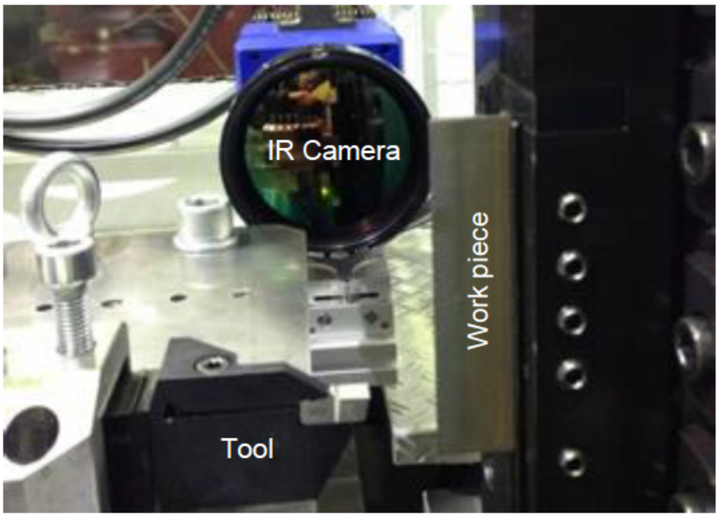

FIGURE 3.1 – Experimental setup (AUGSPURGER *et al.*, 2016a)

An important factor for a reliable temperature measurement is the correct choice of the components' emissivity. To ease the emissivity determination, both the tool and the workpiece were coated with a black ink, allowing the emissivity evaluation for this case, which provided a value of $\epsilon = 0.85$. It is also important to highlight that the camera settings, factors as integration time and filters are also essential to determine a reliable measurements due to the amount of electromagnetic radiation received on the camera's sensors. The higher are the temperatures, higher is the energy produced and smaller should be the integration time, which is the time taken by the energy sensor to receive radiation and then convert it into temperature. The configurations were made to allow measurements in a range from 200 ^{o}C to 900 ^{o}C.

The tool material was uncoated carbide insert (Sandvik H13A) with rake angle of 6^{o}, clearance angle of 3^{o}, cutting radius $r_{\beta} \leq 5\mu$m and width 4.4 mm. The workpiece material was AISI 1045 normalized steel and its dimensions were 3.5 x 200 x 80 mm width, length and height, respectively. For the given range of temperature, the thermal conductivity was estimated to be $k = 75.4W/mK$ and for tool heat capacity a regression function was built ($c(T)$) for corresponding temperature and heat capacity (equations 3.1 and 3.2).

For force acquisition during the process a three-component piezoelectric force platform was used, determining the cutting force and passive force. Since the cutting process was carried out in a linear and constant motion, it is possible to determine the overall power P with velocity and cutting force. From the values obtained of forces along the cutting process, a mean value was taken to be used on power calculation, equation 3.3.

All the experiments were held without coolant, with cutting speed of 100 m/min and 150 m/min and a_p = [0.2, 0.3, 0.4, 0.5] mm (table 3.1).

The analysis method was built on MATLAB platform with the support of its image processing toolbox. FLIR software has a way of exporting the thermal images direct to .mat format, which are matrices projected to MATLAB environment. Each pixel from the exported images contains information about its position and temperature.

Experiments	Cutting Velocity [m/min]	Uncut chip thickness [µm]	Integration time [µs]	Cutting Force [N]	Passive Force [N]	Heat treatment
VP41_1_H200_V100_C45_MF_425	100	200	425	1500	1000	Normalized
VP41_2_H200_V100_C45_MF_425	100	200	425	1565	1005	Normalized
VP42_1_H300_V100_C45_MF_425	100	300	425	2250	1159	Normalized
VP42_2_H300_V100_C45_MF_285	100	300	285	2136	1079	Normalized
VP43_1_H400_V100_C45_MF_285	100	400	285	2716	1118	Normalized
VP45_2_H200_V150_C45_MF_425	150	200	425	1448	688	Normalized
VP46_1_H300_V150_C45_MF_285	150	300	285	2006	801	Normalized
VP46_2_H300_V150_C45_MF_285	150	300	285	2004	875	Normalized
VP49_1_H400_V150_C45_MF_285	150	400	285	2675	1046	Normalized
VP49_2_H400_V150_C45_MF_285	150	400	285	2590	1000	Normalized
VP50_1_H500_V150_C45_MF_285	150	500	285	3220	1120	Normalized
VP50_2_H500_V150_C45_MF_285	150	500	285	3178	1162	Normalized

TABLE 3.1 – Design of experiments (AUGSPURGER *et al.*, 2016a)

As a machining process, the orthogonal cutting performance is subjected to many parameters like workpiece material, shape of tool, depth of cut and others. Because of it, the developed algorithm needs information about all these parameters to work as close as possible to real conditions. Then, all the necessary input data can be summarized on the following table:

Inputs					
Tool		Camera		Workpiece	
Heat Conductivity [W/(mK)]	75,4	Pixel pitch (Infrared Camera) [mm/pixel]	0,015	Length of the workpiece [mm]	200
Heat Capacity [J/(cm^3K)]	Interpolation*	Maximum digit level valid (FLIR X)	8192	Heat Capacity [J/(kgK)]	Interpolation**
Rake Angle [°]	6	Maximum digit level valid (FLIR SC7600)	16000	Workpiece Material	AISI 1045 (normalized)
Clearance Angle [°]	3	Frame Rate (Infrared Camera) [Hz]	328	Width [mm]	3,5
Cutting edge radius [µm]	< 5	Minimum valid temperature for the frames [°C]	200	Percentage of the deformation energy converted into heat	0,9
		Emissivity (Experimentally determined - tool and workpiece coated)	0,85	Density (based on steel) [kg/m^3]	7874

TABLE 3.2 – Algorithm inputs (AUGSPURGER *et al.*, 2016a)

The heat capacities of tool and workpiece materials are used as an interpolation function on the code, using data provided on tables 3.3 and 3.4. The functions are given by the following equations:

$$c_p^T = 2.51 \times 10^{-10} \times T^3 - 1.99 \times 10^{-6} \times T^2 + 0.0027 \times T + 3.09 \tag{3.1}$$

$$c_p^W = -4.39 \times 10^{-7} \times T^3 - 7.07 \times 10^{-4} \times T^2 + 0.0489 \times T + 481.21 \tag{3.2}$$

Workpiece Material			
Temperature [°C]	Heat Capacity [J/(kgK)]	Heat Conductivity [W/(mK)]	Density [Kg/m³]
20	474,62	48,03	7820,9
100	487,94	47,21	7794,3
200	501,2	45,82	7764,2
300	521,29	42,74	7732
400	545,69	39,1	7697,4
500	572,7	35,35	7660,4
600	601,83	31,73	7620,9
700	632,89	28,33	7578,7
800	696,29	23,52	7579,4
900	693,79	25,25	7528,3
1000	691,3	26,61	7475
1100	688,81	27,9	7419,7
1200	686,34	29,34	7362,3

TABLE 3.3 – Workpiece material data (AUGSPURGER *et al.*, 2016a)

Tool Material		
Temperature [°C]	Heat Capacity [J/(cm³K)]	Heat Conductivity [W/(mK)]
20	3,2	100
100	3,24	94,8
200	3,59	88,3
300	3,79	81,9
400	3,9	75,4
500	3,97	68,9
600	4,05	66,7
700	4,14	64,8

TABLE 3.4 – Tool material data (AUGSPURGER *et al.*, 2016a)

3.2 Methods

3.2.1 Power calculation

It is assumed that all mechanical work produced is used to generate heat in the primary shear zone (ABUKHSHIM *et al.*, 2006). Hence, given the cutting force on the movement direction and the cutting velocity, the overall power produced and converted into heat is stated on equation 3.3.

$$P = F_c v_c \tag{3.3}$$

3.2.2 Thermal enegy - chip and tool

The methods used in this paper to calculate the heat flow through the tool and the energy carried away by chip are based on (BOOTHROYD, 1963).

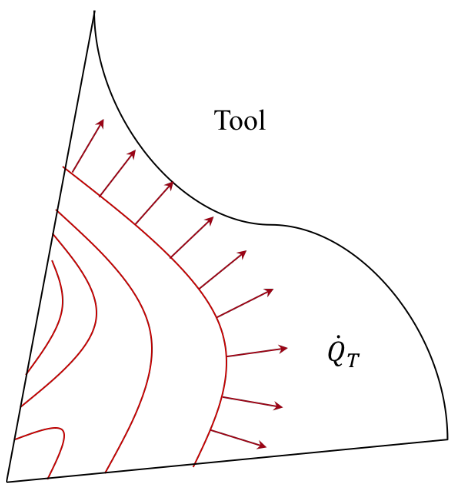

FIGURE 3.2 – Heat flow through tool

Besides the temperature matrix, to calculate heat flow through the tool the following parameters are needed: the heat conductivity, the length of the chosen isothermal line, the temperature gradient normal to this isotherm and the width of the tool. The calculation is given by the following equation:

$$\dot{Q}_T = kL\frac{dT}{dz}w \tag{3.4}$$

For the energy carried away by the chip when it is flowing through control volume, the variables necessary to calculate this value are the heat capacity function $c_p(T)$ of workpiece, the chip temperature distribution along the line where the chip loses contact with tool T_C^{out}, the environment temperature T_e, the velocity of the chip normal to the line of end of contact v_{chip}, the chip thickness t_C and the chip width w.

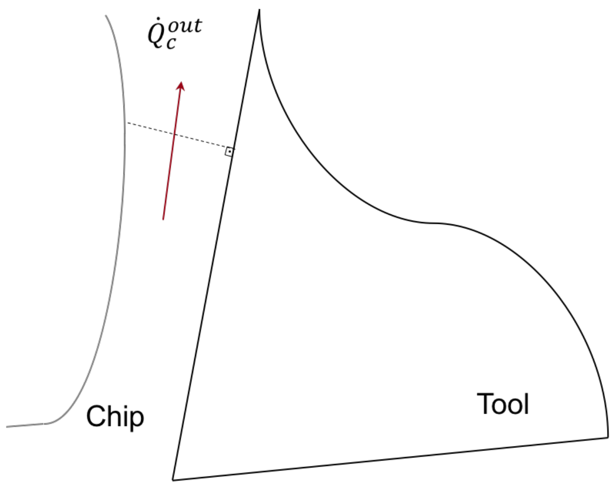

FIGURE 3.3 – Thermal energy carried away by chip

The equation for this energy is represented below:

$$\dot{Q}_C^{out} = c_p^W(T_C^{out} - T_e)v_{chip}t_C w \tag{3.5}$$

In this way, having the location and the temperature of each pixel related to the isotherms and the line of end of chip-tool contact, the math necessary to perform these equations is simple, providing reliable outcomes.

3.2.3 Volume control

For a matter of validation of the presented method and the lack of measurable temperatures on the workpiece surface, the control volume on figure 3.4 was designed.

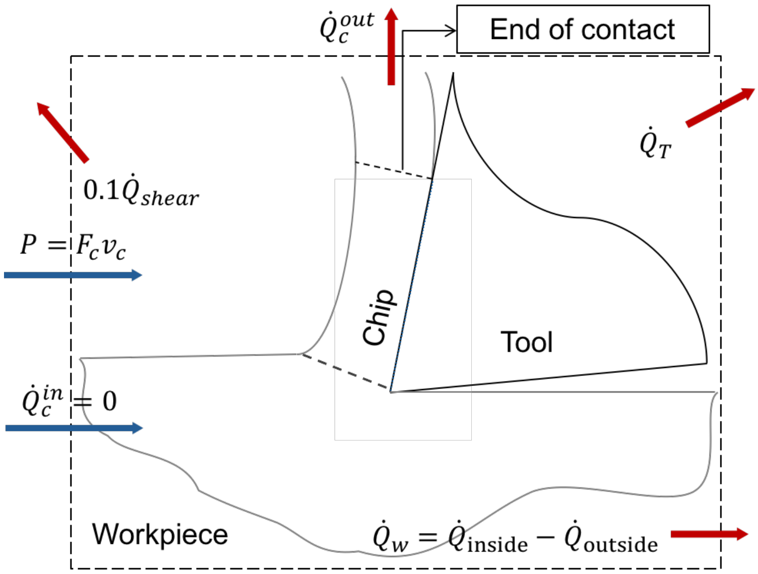

FIGURE 3.4 – Control volume

The shear energy used to raise the temperature of the heat zones is calculated by means of equation 3.6

$$\dot{Q}_{shear} = F_c v_c - F_p v_{chip} \tag{3.6}$$

It is estimated that 90% of this energy generated in the primary shear zone ($\dot{Q}_{shear}$) is converted into sensible heat (TRIGGER; CHAO, 1942). The others 10% are soon dissipated out the control volume. Thus, the energy balance of the control volume will provide:

$$\dot{Q}_W = P - \dot{Q}_T - \dot{Q}_C^{out} - 0.1\dot{Q}_{shear} \tag{3.7}$$

4 Results

4.1 Code implementation

4.1.1 MATLAB environment

As mentioned on chapter 3, FLIR software provides indexed matrices in .mat format to be used directly in MATLAB. Each pixel contains the temperature information about that location in the picture. It is possible to visualize an example of a scaled image on the following figure:

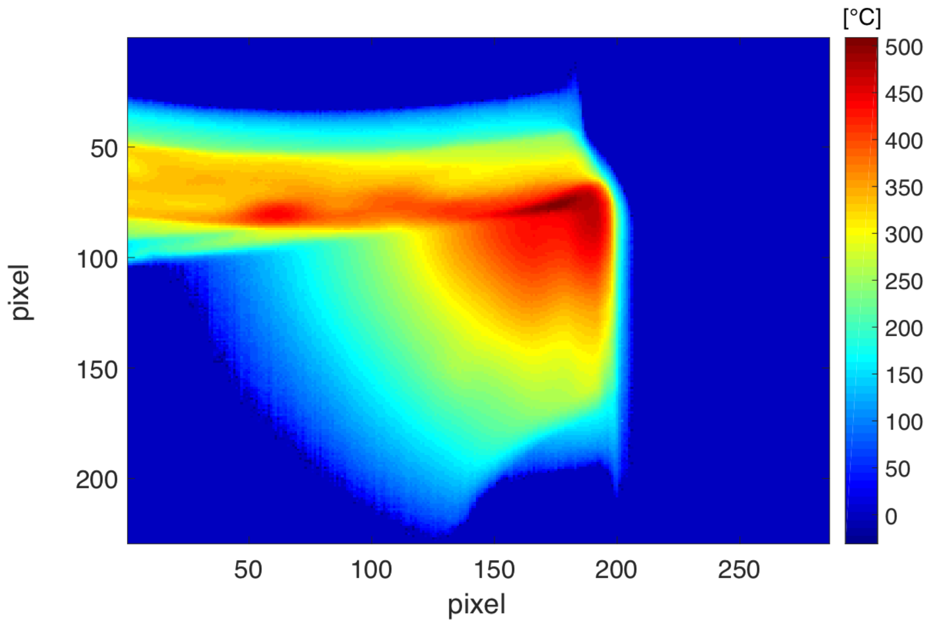

FIGURE 4.1 – Thermal image for $t_{uc} = 200\mu m$ and $v_c = 100m/min$ scaled in MATLAB

From figure 4.1 and with MATLAB Image Processing Toolbox support it was possible to extract features that will futher help to capture the behavior of heat flows and heat partitions. They are:

- Rake and clearance face recognition

- Detection of tool tip
- Image segmentation of tool, chip and workpiece
- Isotherm coordinates along tool shape

4.1.2 Auxiliary functions

Some functions used to build the method were already implemented in MATLAB's library. To understand the output of these functions, the next subsections will present their operation and final aim.

4.1.2.1 Contour plot

An important tool for the development of the method, contour plot is able to provide same level curves. Since the basic variable provided is the temperature along cutting zone, this function will calculate continuous lines of very close values of temperature. Doing it with a small tolerance, the lines calculated will be the corresponding isotherms of the image. Hence, it is easier to extract the coordinates of each pixel in these lines for each level of temperature.

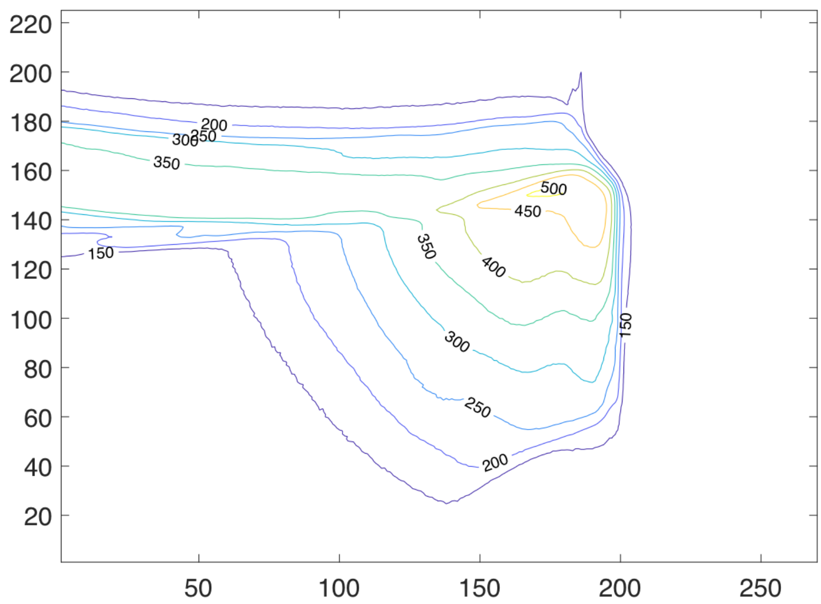

FIGURE 4.2 – Contour plot for $t_{uc} = 200\mu m$ and $v_c = 100m/min$

4.1.2.2 Hough lines transformation

Hough transform is an extensive method used in computer vision. It is an extraction feature for complex geometries, using normal parameterization for straight lines (DUDA; HART, 1972). In MATLAB, the parametric equation of the lines is stated in equation 4.1.

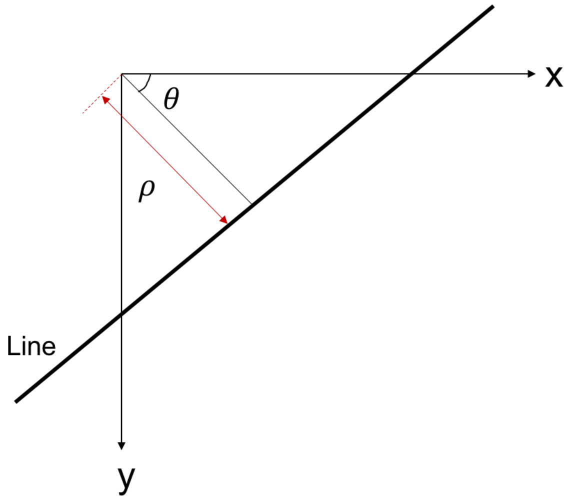

FIGURE 4.3 – Reference for hough parametrization (Source: Mathworks)

$$\rho = x \times cos(\theta) + y \times sin(\theta) \tag{4.1}$$

Concerning the images, the rake and clearance face can be mapped by means of hough lines transformation in MATLAB. It is necessary to provide a probable range of angles in which the angular coefficient of the sought lines is defined. The more precise is this range, more reliable and faster will be the output.

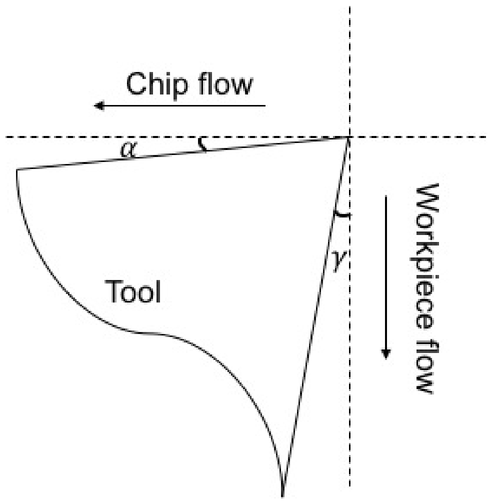

FIGURE 4.4 – Placement of tool

The test bench, where the experiments were performed, allows for a fixed positioning of the tool in relation to the thermal camera. It means that the angle between the rake face and horizontal line and the angle between clearance face and vertical line are always the designed rake and clearance angles, respectively. In other words, the tool does not rotate in relation to the reference axes. Because of this, it is possible to perform hough transformation on the image, with very high accuracy. However, sometimes the chip can cause interference on the measurements of the edges, making the feature extraction impractical.

4.1.3 Implementation steps

This subsection will present an overview of the code implementation and the logical sequence of what was implemented, showing each step taken to develop the next one. The steps that will be presented represent the key path for the code development.

4.1.3.1 Overview

The program was able to identify tool and chip shapes, allowing the image segmentation of the components and, consequently, the thermal analysis of each part separately. By means of image processing and input data about cutting parameters, features like maximum temperature in the cutting zone, maximum chip temperature, heat flow through chip and tool are some examples of what the code is able to provide.

4.1.3.2 Finding tool edges

As mentioned in the subsection Hough lines transformation, the Hough transform was essential to detect edges. In order to make the code faster, the edges of the original thermal image were extracted, creating a binary image (figure 4.5).

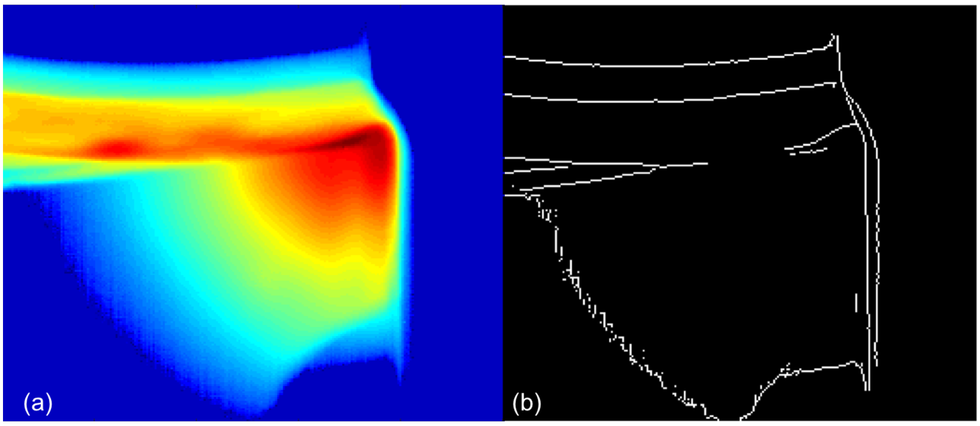

FIGURE 4.5 – (a) Original thermal image and (b) Edges detected by MATLAB

Given the previous figure, the hough transform is applied directly on the binary image, which only has information about the edges. Because of it, the hough transform performance is faster than it would if applied to the original image. The source code for hough implementation is showed on the following script:

```
1         function obj = calculateCoordinates(obj)
2             obj.BW = edge(obj.frame,'sobel');
3             imshow(obj.BW)
4             figure
5             imagesc(f.*(ones(size(f))-obj.BW))
6             %-------------------------Finding the clearance face-----------
7             [H, THETA, RHO] = hough(obj.BW,'Theta',2:5);%Hough transformation
8             %used to find angles between 2 - 5
9             P  = houghpeaks(H, 10);
10            obj.lines = houghlines(obj.BW, THETA, RHO, P, 'FillGap', 15,'MinLength',10);%Here we can find the lines of cutting edge and afterwards find the coordinate of the tool tip
11            l = length(obj.lines);
12            obj.coordCF = [];
13            for i=1:l%Loop to test the resultant lines from the command houghlines
14                Theta=obj.lines(i).theta;
15                t1 = obj.lines(i).point1;
16                t2 = obj.lines(i).point2;
17                rho = obj.lines(i).rho;
18                imagesc(f)
19                hold on
20                plot([t1(1) t2(1)]',[t1(2) t2(2)]','m')
21                hold off
22                if rho < 204 && rho > 198
23                    obj.coordCF = [t1;t2];
24                    obj.ClearanceAngle = Theta;
25                end
26            end
27            %-------------------------Finding the rake face----------------
28            [H, THETA, RHO] = hough(obj.BW,'Theta',81:85);%Hough transformation
29            %used to find angles between 5 - 9 (must be the complementary on this direction)
30            P  = houghpeaks(H, 10);
31            obj.lines = houghlines(obj.BW, THETA, RHO, P, 'FillGap', 15,'MinLength',10);%Here we can find the lines of cutting edge and afterwards find the coordinate of the tool tip
32            l = length(obj.lines);
33            obj.coordRF = [];
34            for i=1:l
35                Theta=obj.lines(i).theta;
36                t1 = obj.lines(i).point1;
37                t2 = obj.lines(i).point2;
38                rho = obj.lines(i).rho;
39                imagesc(f)
40                hold on
41                plot([t1(1) t2(1)]',[t1(2) t2(2)]','m')
42                hold off
43                if rho < 103 && rho > 98
44                    obj.coordRF = [t1;t2];
45                    obj.RakeAngle = 90 - Theta;
46                end
47            end
48        end
```

Since the rake angle is 6^o and the clearance angle is 3^o, ranges of [81:85] and [2:5] were given to each one respectively, as it can be observed on lines 4 and 20. Regarding the rake angle, the range of angles is given by the complementary operation due to the reference in hough method. This way, the hough transform returns highlighted points in the accumulation matrix of hough process and from them the 10 first points are chosen to be analyzed, which is a reasonable amount of points which may represent sections of the rake and clearance lines.

The fixed position of the tool also allows the predetermination of the ρ parameter, which is the distance of the detected lines from the reference in hough. This is also seen on lines 14 and 30 as boundary conditions to determine the right edge lines. The outputs of this function are the extremity coordinates of the detected line and also the angle of the corresponding angular coefficient.

Sometimes it was necessary to set default conditions, as rake and clearance angles, due to chip obstruction. The chip obstruction causes interference on the temperature fields, which can disturb the edges definition in thermal images, preventing the proper functioning of hough transform.

4.1.3.3 Rake and clearance face

With the data provided by the output of the hough function, the equation that defines each straight line corresponding to the edges is known. So, it is possible to extend the lines to match the entire rake and clearance edges.

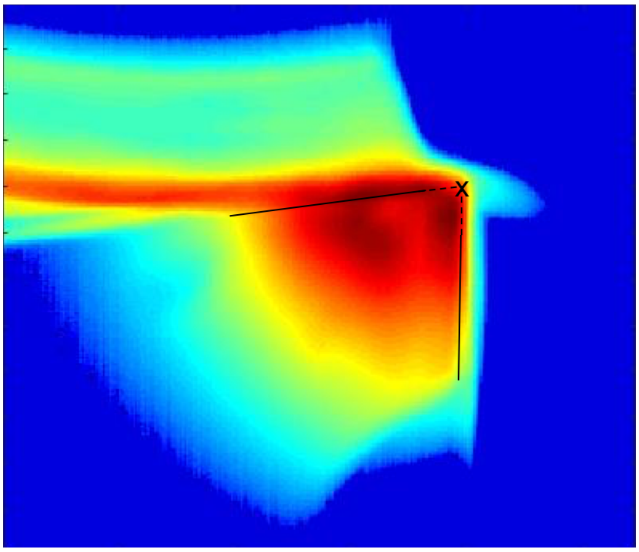

FIGURE 4.6 – Lines detected by hough transformation method

This is an important step of the method because it allows to build a mask (figure 4.7), which is a binary image with the tool shape, that is able to remove only the region of interest. Consequently, it will be possible to analyze the temperature fields and thermal behavior inside the tool without any interference from the temperatures in the vicinity.

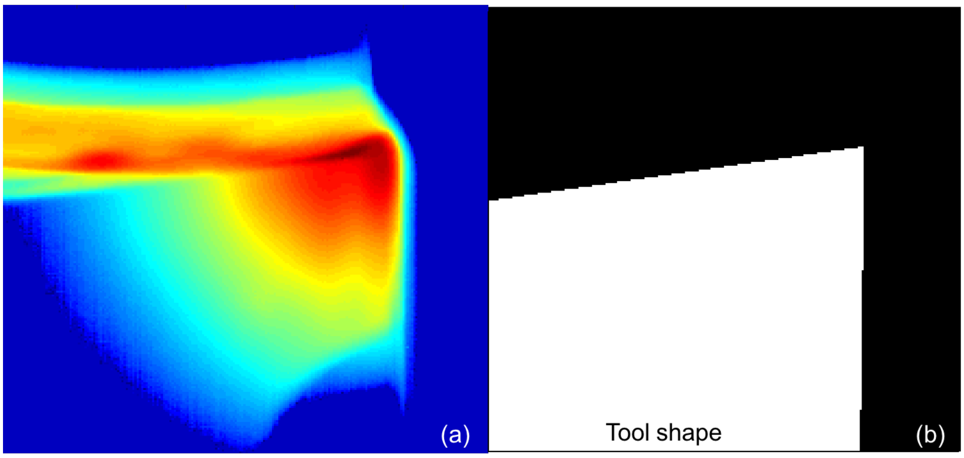

FIGURE 4.7 – (a) Original image and (b) Region of interest - tool

4.1.3.4 Tool tip coordinates

As the rake and clearance edges are determined, the tool tip is calculated by means of the intersection between these lines. On the figure 4.6, the found lines are extended until they intersect, then the tool tip coordinates can be calculated. It is important to determine these coordinates due to the interest in knowing the temperatures of the area close to the tip and what is the maximum value it can reach, which is related directly with tool life and therefore the surface finish.

4.1.3.5 Maximum temperatures

Since the code was able to segment the tool shape from the entire matrix, it gets easier to extract the chip contour, which is the other region of interest which presents measurable range of temperatures. Getting the maximum temperature from each zone allows not only to know if the measured temperatures are inside the measurement limit, but also to compare the behavior of this maximum temperature for different cutting velocities and depths of cut.

The maximum temperature in the cutting zone raises as the cutting process happens (figure 4.8). For a same value of underformed chip thickness, the higher is the cutting velocity the lower will be the value of maximum temperature reached.

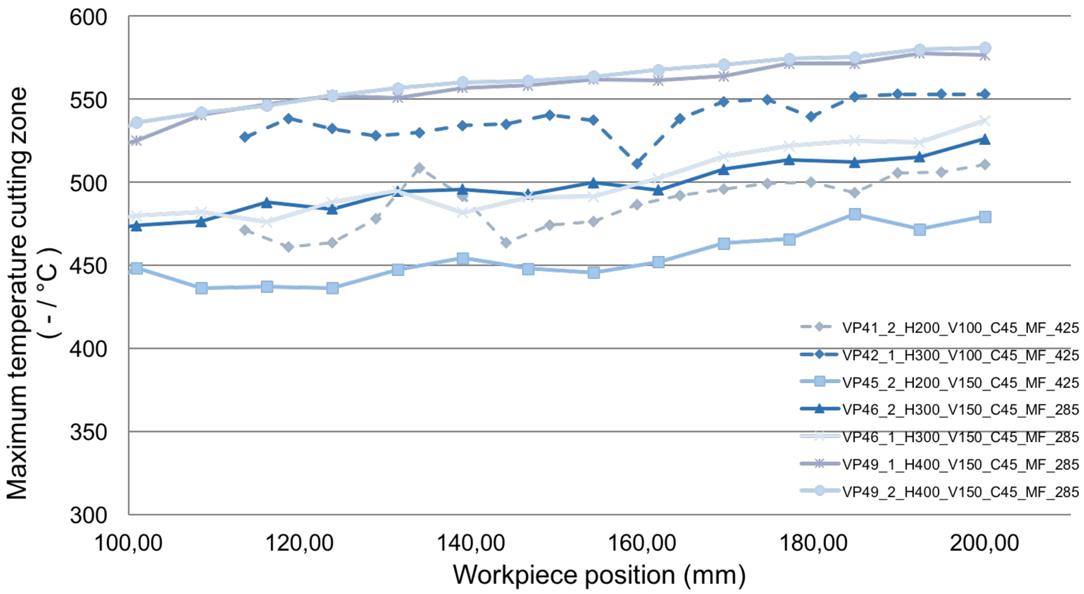

FIGURE 4.8 – Maximum temperature of the cutting for the designed experiments

4.1.3.6 Temperature fields

In this step, the auxiliary function will be used mentioned on the subsection Contour plot. For the application in this method, a step of 40^oC was taken to separate each level of temperature. The use of the function is simple, being necessary only to provide the image (`obj.frame`) and the spacing vector between levels (`v`).

```
[C,~] = contour(obj.frame,v);%Getting the contour lines for C
```

The output of contour function is a matrix C with 2 rows that will provide the levels of temperature and the number of coordinates followed by their absolute values of `x` and `y`, providing the location of each point of the isothermal curves.

```
C     =  [C(1) C(2) C(3) ...C(k)... C(N)]
C(k)  =  [level x(1) x(2)...
          numxy y(1) y(2)...]
```

For each matrix `C(k)`, `level` shows which temperature the following points are representing and `numxy` is the number of coordinates used to build the corresponding level. The coordinates are represented in the pair `(x,y)`.

Also, it is possible to visualize the evolution of the isotherms and, consequently, the temperature gradients with contour plot. An example can be observed on figure 4.9. The distance between the lines slightly increases as the cutting process advances. This points to a reduction of the gradient values, which indicates a negative rate of heat into the tool along the time.

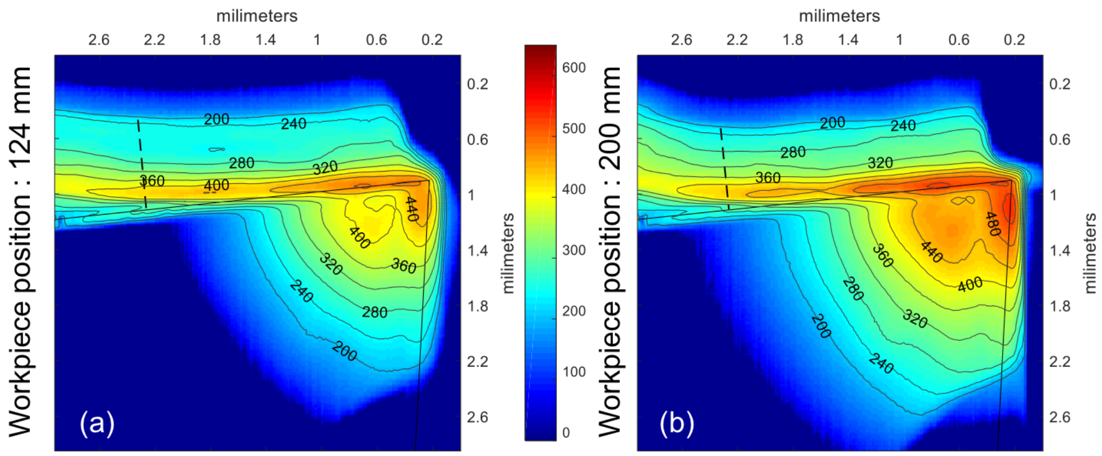

FIGURE 4.9 – (a) Temperature field for workpiece position 124 mm (b) Temperature field for workpiece position 200 mm

4.1.3.7 Heat flows - Chip and Tool

As described on section 3.2, the heat flow through the tool and the energy carried away by the chip are calculated. For heat flow through the tool, it is possible to extract isothermal lines by means of contour plot and to calculate the temperature gradient, which is already normal to the isothermal lines due to its properties. The tool width is already known (subsection 3.1). The length of the chosen isotherm is given by counting the amount of pixels in `numxy`, as described in the previous subsection, and then it is converted to millimeter with the scale factor. In the case of the energy carried away by chip, the chosen line is placed on the end of chip-tool contact, which is where the maximum temperature of the chip - tool interface occurs (ABUKHSHIM *et al.*, 2006), (BOOTHROYD, 1963). The explanation is that all the heat source in the friction zone is located before this line. In other words, there is no other heat source after this line that could provide more thermal energy to be carried away by chip.

4.1.3.8 Heat partitions

Having the results of the subsection 4.1.3.7, these values can be combined with the total power (P) generated during the cutting process (equation 3.3) to calculate the energy that goes to the workpiece by means of energy balance (equation 3.7). Hence, it is possible to calculate the heat partition relative to each zone of interest.

$$p_i = \frac{\dot{Q}_i}{P} \tag{4.2}$$

Where the index i is related to C (chip), W (workpiece) and T (tool).

4.2 Method validation

As described along section 4.1, there are many outputs from the implemented method, shear and normal stresses related to the mechanical part, for example. However, in this paper the heat partitions will be the focus of discussions.

The total power produced along this high speed machining was calculated as in the equation 3.3. The values are shown on figure 4.10.

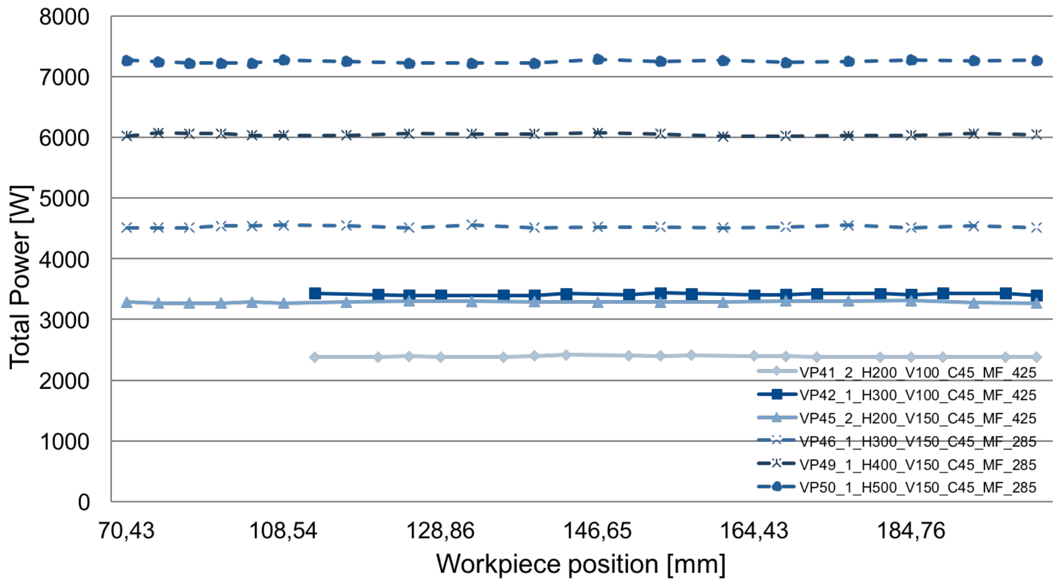

FIGURE 4.10 – Total power produced

As expected, the higher are the values for cutting velocity or depth of cut, higher are the values for total power produced. For each experiment, the computational method was able to provide the thermal energy that goes to tool, chip and workpiece by means of energy balance. Then, the thermal behavior of every area of interest along the workpiece position can be observed. The measurement starts when a reasonable area of the cutting zone reaches the minimum measurable temperature. For cutting velocity of $150m/min$ it starts earlier because the rate of heat production is higher than when the cutting velocity is $100m/min$.

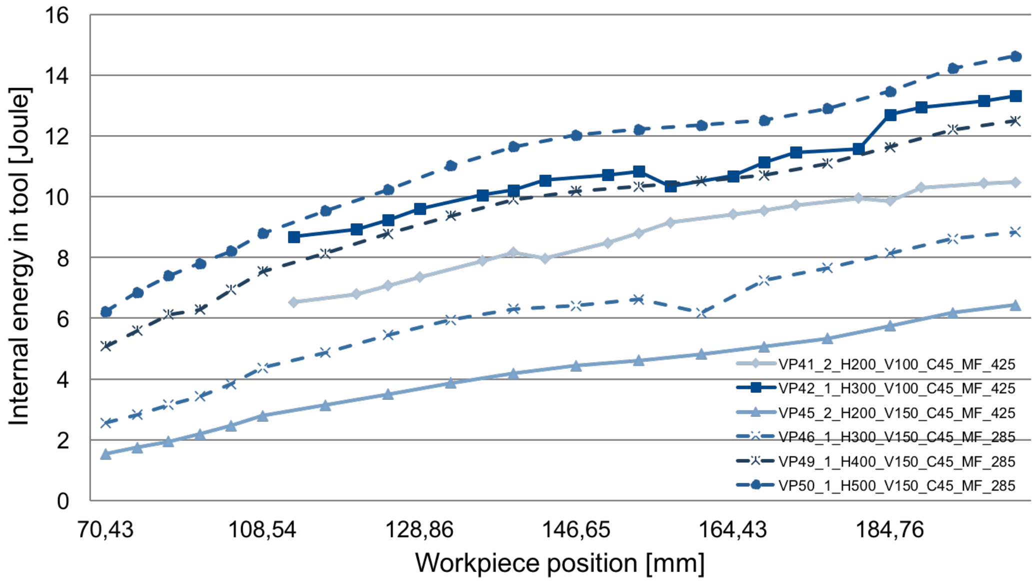

FIGURE 4.11 – Inner energy of the tool along workpiece position

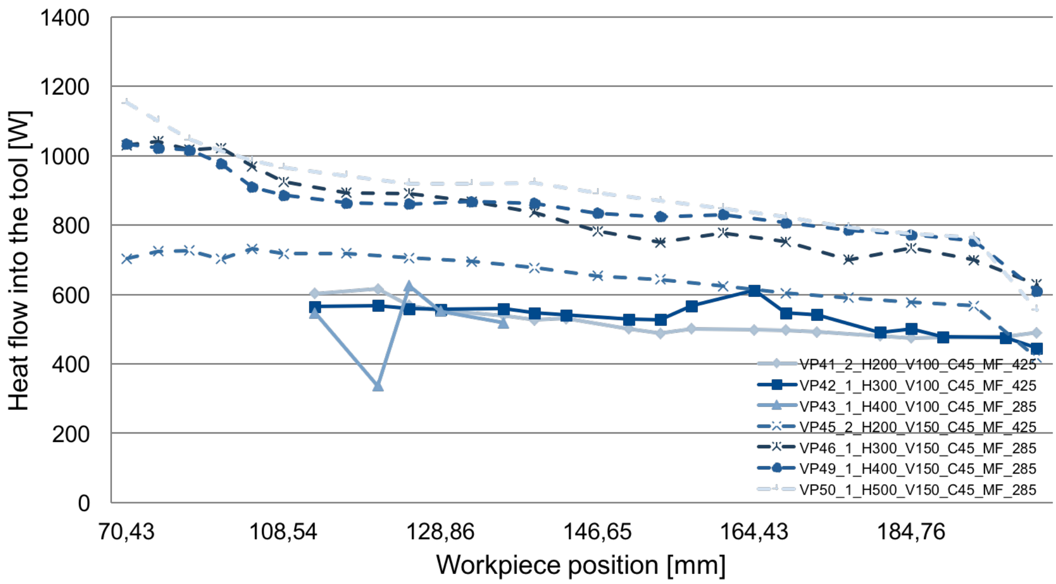

FIGURE 4.12 – Heat flow into the tool

As it can be observed on the previous figure 4.12, the change rate of the inner energy of the tool begins with a higher value than in the end of the process. The rate starts to stabilize, indicating the beginning of the steady state.

Still regarding the tool (figure 4.13), the partition of energy can reach a range that goes from about 20% in the transient state down to 4% close to the end of the cutting process, where it reaches the steady state. Takeuchi *et al.* (1982) presents that 10 - 30% of the total heat generated is removed through the tool, which is in agreement with the presented result.

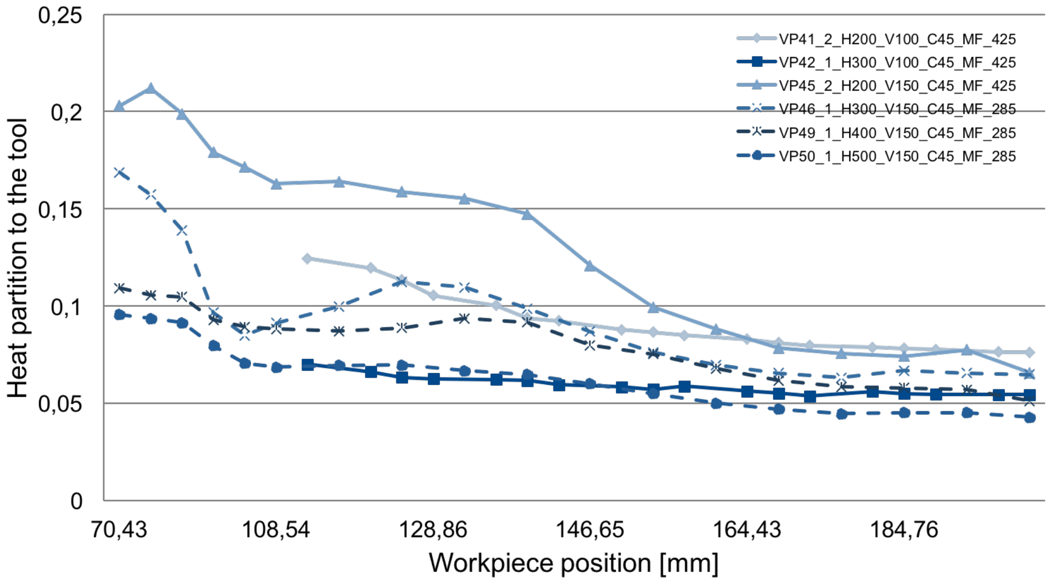

FIGURE 4.13 – Heat partition ratio for the tool

Also Augspurger *et al.* (2016b) proposed a model based on the use of Green's function for temperature prediction along the tool shape during transient state. The experiments were the same as the ones presented in this work. Using the same heat flow into the tool to simulate temperature fiels, the result showed a good approximation between model and experimental data.

Regarding the chip, it is important to highlight the total power produced during the cutting process, which has a significant value because of the high values of cutting velocity and force. Moreover, it must be also noticed the amount of energy that goes to the chip (figure 4.14 and 4.15). The chip takes around 70% of the total energy produced, which may be explained by the high temperatures that the region can reach and the high velocity of flowing.

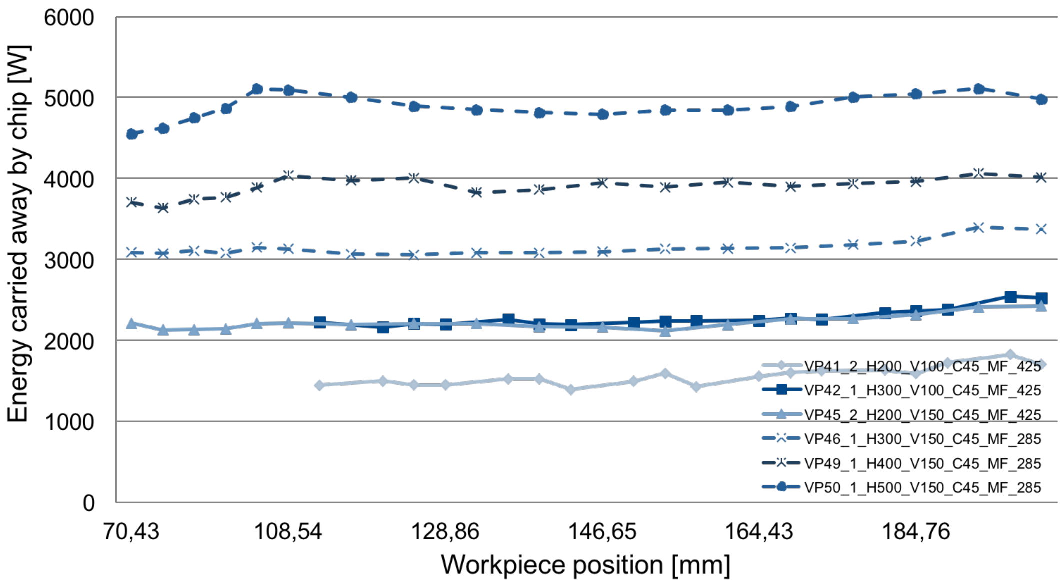

FIGURE 4.14 – Thermal energy into the chip

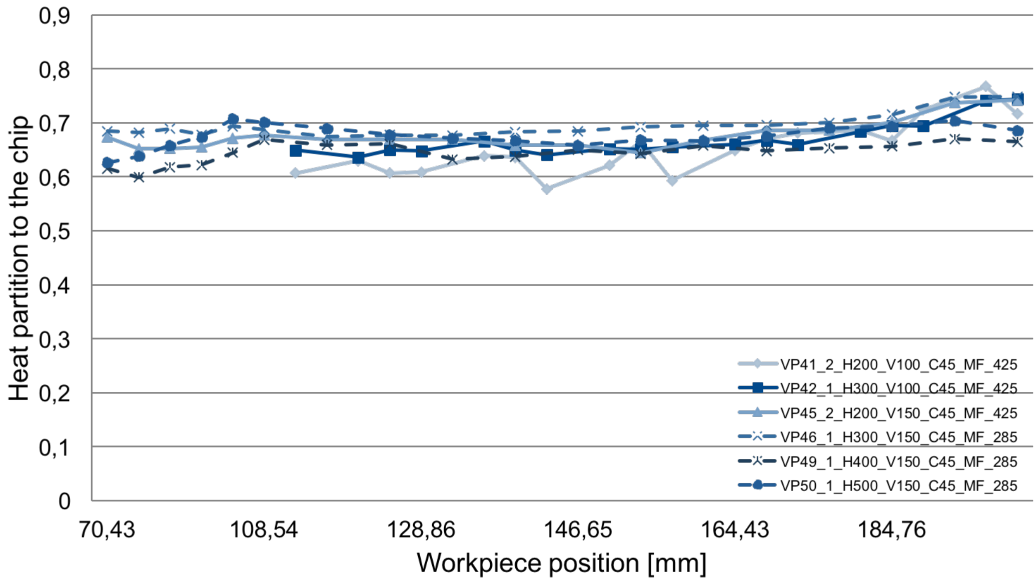

FIGURE 4.15 – Heat partition ratio for the chip

Trigger e Chao (1942) presented a model for the average temperature prediction in the cutting zone. The proposed model was made for steady state and the existence of only two heat sources was considered, the primary and secondary shear zones. The heat partitions were assumed as 90% for the chip and 10% for the tool. It is known that the tertiary zone has a minor influence on heat generation given the cutting conditions (subsection 2.1.1) and, consequently, the temperature raise in the tool is mainly caused by the friction zone. Then, the heat partition about 70% for the chip (figure 4.15) and 6% for the tool (figure 4.13) in the steady state will represent around 92% for the chip and 8% for the tool of all heat produced in the primary and secondary shear zones. It agrees with the proposed assumption.

It may be noticed that for experiments with the same relation $v_c \times a_p$ (figure 4.14) apparentely the same amount of energy is carried by the flowing chip. On the other hand, the heat flow into the tool is lower as the cutting speed is higher, which can be analyzed to obtain better conditions to increase tool life. However, since the thermography method is very sensible to external interference and many experiments were affected as mentioned before, it would be necessary to perform new experiments to validate this hypothesis.

To exemplify the results, the experiment with cutting velocity $v_c = 150 m/min$ and depth of cut $a_p = 500 \mu m$ will be taken to represent the behavior of the outcomes regarding heat partition. All others experiments had approximately the same behavior during the cutting process.

Concerning the heat partition along the tool, the workpiece and the energy carried away by the chip, their behaviors can be observed on the figure 4.16. There is a slight decrement in the heat flow through the tool, which was expected due to the steady state as discussed before. As for the energy carried by the chip, a slight increment may be noticed.

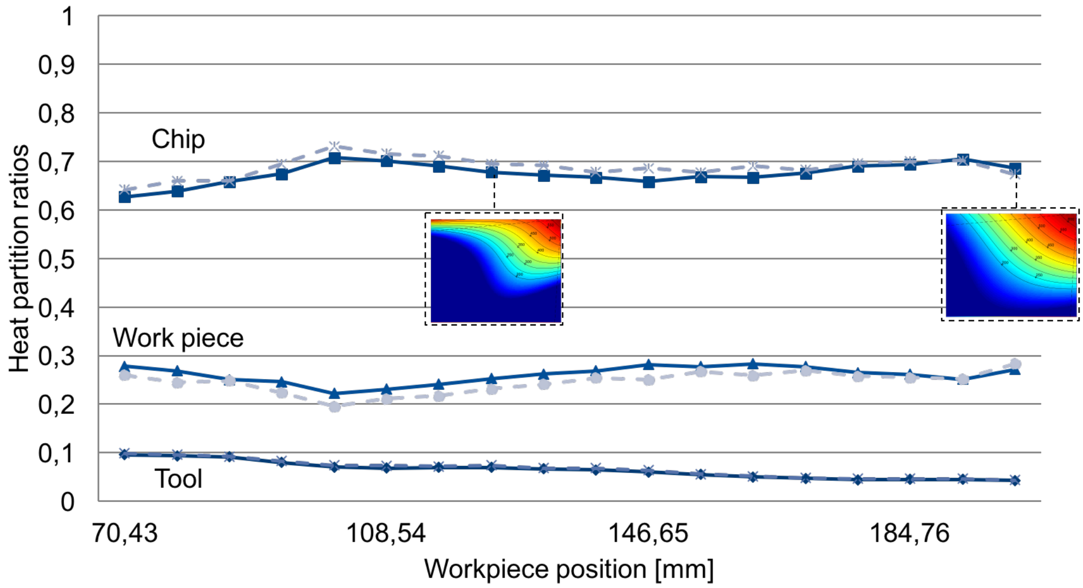

FIGURE 4.16 – Heat partition for experiment with $a_p = 500\mu$m and $v_c = 150$ m/min

5 Conclusions

The computational method had a simple implementation, requiring the support of the image processing toolbox of MATLAB. The combination of a high speed camera with the fast processing of thermal images provided a tool for fast inspection of orthogonal cutting. The results were obtained for AISI 1045 steel, but it can be easily adapted for other types of materials by providing the new cutting parameters inside the source code.

The results found when processing thermal images provided a reasonable understanding about heat distribution throughout tool and chip components. Most part of the data provided by the cutting process regards to transient state, but is also possible to note it reaching the steady state close to the end of the cutting process, which this computational method is also able to analyze for this part of the process. It can be a valuable tool for validation of future studies concerning heat sources modeling and simulation.

Most of the heat generated during the cutting process is removed by the chip, about 70% of the total generated power. The steady state showed a partition for the tool of about 6%, which agrees with the model proposed by Trigger e Chao (1942). The chip and the workpiece have bigger heat partitions due to their high flowing velocity, while the velocity of heat conduction inside the tool is much smaller.

One of the problems to elaborate this work was that many of the videos were damaged due to chip obstruction interfering on the ideal visualization of each thermal frame. Pieces of chip with different temperatures were captured on tool surface, disturbing the field of temperatures along the tool shape. This fact made it impossible to use some frames from the same video and sometimes entire experiments.

The thermography method for temperature measurement still presents some challenges, mainly when it comes to setting the correct emissivity. Even when coating the tool and the workpiece with black ink and conductiong experiments to determine the its emissivity, the ink cracks close to the tool tip and along the chip. This fact can be a source of error providing an overestimation of the emissivity value and consequently an underestimation of the real temperature. But even taking a reasonable effort to determine the right emissivity for accomplishing a reliable measurement, the termography is still a powerful tool for inspection, specially for cutting processes as discussed in this paper.

With a camera filter capable of measure temperatures lower than 200 Celsius degrees, it would be possible to complete the study with the measurement of temperatures on the workpiece area, providing more results.

Computer vision, as well as image recognition patterns and image processing, is being used each time more in nowadays processes. For a future study beyond the scope of this paper, computer vision can become an even stronger tool when combined with machine learning, which is revolutionizing the most diverse areas. The principles used to build this computational method could be converted to analyze others types of cutting processes, such as milling. Then, it could be turned into an intelligent system to support machining processes, improving all cutting parameters in order to obtain higher tool efficiency, increasing tool life, improving surface finishing of the workpiece and reducing cutting time.

Finally, it is also important to highlight that the method can be developed in other programming languages, such as Python and C++. Although the FLIR software has a direct connection with MATLAB, some programming languages do not require a paid lincense, which can be very interesting for making the solution cheaper.

Bibliography

ABUKHSHIM, N.; MATIVENGA, P.; SHEIKH, M. Heat generation and temperature prediction in metal cutting: A review and implications for high speed machining. **International Journal of Machine Tools and Manufacture**, Elsevier, v. 46, n. 7, p. 782–800, 2006.

AUGSPURGER, T.; KLOCKE, F.; DÖBBELER, B.; BROCKMANN, M.; LIMA, A. **Experimental investigation of temperatures and heat flows for orthogonal cutting 1045 steel by thermal imaging**. 08 2016. 6 p.

AUGSPURGER, T.; KLOCKE, F.; DÖBBELER, B.; BROCKMANN, M.; GIERLINGS, S.; LIMA, A. **Modelling of Transient Thermal Conditions in Cutting**. 08 2016. 8 p.

BOOTHROYD, G. Photographic technique for the determination of metal cutting temperatures. **British Journal of Applied Physics**, IOP Publishing, v. 12, n. 5, p. 238, 1961.

BOOTHROYD, G. Temperatures in orthogonal metal cutting. **Proceedings of the Institution of Mechanical Engineers**, SAGE Publications Sage UK: London, England, v. 177, n. 1, p. 789–810, 1963.

DUDA, R. O.; HART, P. E. Use of the hough transformation to detect lines and curves in pictures. **Communications of the ACM**, ACM, v. 15, n. 1, p. 11–15, 1972.

JEON, J.; KIM, S. Optical flank wear monitoring of cutting tools by image processing. **Wear**, Elsevier, v. 127, n. 2, p. 207–217, 1988.

KHALIFA, O. O.; DENSIBALI, A.; FARIS, W. Image processing for chatter identification in machining processes. **The International Journal of Advanced Manufacturing Technology**, Springer, v. 31, n. 5, p. 443–449, 2006.

KOMANDURI, R.; HOU, Z. B. Thermal modeling of the metal cutting process: part i temperature rise distribution due to shear plane heat source. **International Journal of Mechanical Sciences**, Elsevier, v. 42, n. 9, p. 1715–1752, 2000.

KOMANDURI, R.; HOU, Z. B. Thermal modeling of the metal cutting process part ii: temperature rise distribution due to frictional heat source at the tool chip interface. **International Journal of Mechanical Sciences**, Elsevier, v. 43, n. 1, p. 57–88, 2001.

KURADA, S.; BRADLEY, C. A machine vision system for tool wear assessment. **Tribology International**, Elsevier, v. 30, n. 4, p. 295–304, 1997.

LEBAR, A.; JUNKAR, M.; POREDOŠ, A.; CVJETICANIN, M. Method for online quality monitoring of awj cutting by infrared thermography. **CIRP journal of manufacturing science and technology**, Elsevier, v. 2, n. 3, p. 170–175, 2010.

LEE, S.; NAM, J.; HWANG, W.; KIM, J.; LEE, B. A study on integrity assessment of the resistance spot weld by infrared thermography. **Procedia Engineering**, Elsevier, v. 10, p. 1748–1753, 2011.

MALDAGUE, X. Applications of infrared thermography in nondestructive evaluation. **Trends in optical nondestructive testing**, Elsevier Science, p. 591–609, 2000.

POOLE, N.; SARVAR, F. Fundamentals of heat transfer. In: IET. **Thermal Design of Electronic Systems, IEE Colloquium on**. [S.l.], 1989. p. 1–1.

SARMA, P.; KARUNAMOORTHY, L.; PALANIKUMAR, K. Surface roughness parameters evaluation in machining gfrp composites by pcd tool using digital image processing. **Journal of reinforced plastics and composites**, Sage Publications Sage UK: London, England, v. 28, n. 13, p. 1567–1585, 2009.

SHAW, M. C.; COOKSON, J. **Metal cutting principles**. [S.l.]: Oxford university press New York, 2005.

TAKEUCHI, Y.; SAKAMOTO, M.; SATA, T. Improvement in the working accuracy of an nc lathe by compensating for thermal expansion. **Precision Engineering**, Elsevier, v. 4, n. 1, p. 19–24, 1982.

TRIGGER, K.; CHAO, B. An analytical evaluation of metal cutting temperature. **Transactions of ASME**, v. 73, p. 57–68, 1942.

USAMENTIAGA, R.; VENEGAS, P.; GUEREDIAGA, J.; VEGA, L.; MOLLEDA, J.; BULNES, F. G. Infrared thermography for temperature measurement and non-destructive testing. **Sensors (Basel, Switzerland)**, MDPI, v. 14, n. 7, p. 12305–12348, 2014.

Appendix A - Source Code

A.1 Temperature Analysis

```
classdef TemperatureAnalyze
%TEMPERATUREANALYZE Build an object with general informations about one
%previously determined matrix from an entire struct of these matrices.
%OBJ = TEMPERATUREANALYZE(FRAME,INDEX) returns an object with all the
%properties of this class.

    properties(GetAccess = 'public', SetAccess = 'private')
        CoordinateToolTip;
        TemperatureToolTip;
        RakeAngle;%Rake face slope
        ClearanceAngle;%Clearance face slope
        ShearAngle;
        FrictionAngle;
        MeanTemperatureTool;
        MaximumTemperatureTool;
        MaximumTemperatureChip;
        MaximumTemperatureCuttingZone;
        HeatCarriedAwayByChip;
        HeatFluxAwayFromToolTip;
        HeatFluxThroughWorkpiece;
        TotalPowerBalance;%Total energy produced during the process
        InternalEnergyTool;%Energy only for valid pixels
        CuttingForcePowerDirection;
        CuttingForceUncutChipThicknessDirection;
        CuttingForceParallelToolFace;
        CuttingForceParallelShearPlane;
        CuttingForcePerpendicularShearPlane;
        CuttingForcePerpendicularToolFace;
        CoefficientFriction;
        ShearStress;
        NormalStress;
        PecletNumber;
        RatioR;
        ShearEnergyVolume;
        FrictionEnergyVolume;
        CuttingVelocity;
        UnCutChipThickness;
        ContactLength;
    end

    properties(GetAccess = 'private', SetAccess = 'private')
        coordRF;
        coordCF;
        BW;
        lines;
        frame;%Variable of the single frame that will be analyzed
        pointCF;%auxiliar to plot the cutting edge
        pointRF;
        pointM;%Point on the bissectrix of RF and CF
        Tx;%auxiliar to plot the gradients of the frame x axis
        Ty;%auxiliar to plot the gradients of the frame y axis
```

```
        biImageTool;%Binary image of the tool shape
        biImageChip;
        biShearLine;
        xyMaxTemp;%coordinates of the point inside the chip with maximum Temperature
        lineChip;
        lineTool;%
        validTemperature;%Minimum temperature inside the valid range
        heatCapacityTool;%Heat capacity for the tool(regression of points)
        heatCapacityWP;%Heat capacity for the tool(regression of points)
        heatConductivity;%Heat conductivity for the tool
        nExcPoints;
        heatAccumulatedPerLine;
        ptosLines;
        extPtosLineChip;
        line200;%auxiliar variable to get the last valid isotherm
    end

    methods
    %Here are listed the functions that are used to build the object
        function obj = TemperatureAnalyze(Frame,index)%constructor
            %Inputs -----------------------------------------
            Fp =  Frame.Fp;%Cutting force in the power direction (Newtons)
            Fq = Frame.Fq;%Passive force (Newtons)
            widthTool = Frame.w;%meters
            ap = Frame.ap;%meters
            Vp = Frame.Vp/60;%meters/second
            tuc = Frame.tuc;%meters
            clength = Frame.clength;%contact length tool - chip
            obj.validTemperature = 200;% For any experiment
            A = 0.1;%percentage of the deformation energy that is converted in heat
            obj.heatConductivity = 75.4;%Tool
            obj.heatCapacityWP = [-4.39956806034758e-07 0.000707314520321484 ...
            -0.0488770693887544 481.214007868631]; %AISI 1045
            obj.heatCapacityTool = (10^6)*[2.50542895559373e-10 ...
            -1.99579761670655e-06 0.00274369536032376 3.09265830398264];
            %-------------------------------------------------
            obj.CuttingVelocity = Vp*60;%m/minute
            obj.UnCutChipThickness = tuc;
            obj.frame = Frame.f(index).f;
            obj.ClearanceAngle = Frame.alpha;
            obj.RakeAngle = Frame.gamma;
            if isempty(clength)
                clength = obj.contactLength();
            end
            obj.ContactLength = clength;
            if isempty(Frame.ToolTip)
                obj = obj.calculateCoordinates();
                if ~isempty(obj.coordRF) && ~isempty(obj.coordCF)
                    obj = obj.coordinateToolTip();
                else %Default conditions
                    figure
                    imagesc(Frame.f(index).f)
                    Frame.ToolTip = input('The code was not able to find the tool
    tip position. Provide valid coordinates [x y]: ');
                    close
                    obj.ClearanceAngle = Frames.alpha;
                    obj.RakeAngle = Frames.gamma;
                end
            end
            obj.CoordinateToolTip = Frame.ToolTip;
            obj = obj.toolContour();
            obj = obj.findLineTool();
            obj = obj.chipContour();
            obj = obj.findLineChip();
%             obj = obj.framesOverlap(Frame,index);%used only when
%             necessary
            obj = obj.pointsRFandCF();
            obj = obj.TempTT();
            obj = obj.meanTemperatureTool();
            obj = obj.maxTemperatureTool();
```

```
            obj = obj.maximumTemperature();
            obj = obj.maxTemperatureChip();
            obj = obj.calculateGradient();
            obj = extremePointsChip(obj);
            obj = obj.heatBalance(tuc,Vp,ap,widthTool);
            obj = obj.internalEnergyTool(widthTool);
            obj = obj.shearLine();
            obj = obj.calculatePecletNumber();
            obj = obj.forcesValues(Fp,Fq,ap,tuc);
            obj.TotalPowerBalance = 0.97*(obj.CuttingVelocity*(obj.CuttingForcePowerDirection*(1-A) + obj.CuttingForceParallelToolFace*A*obj.RatioR)/60);
            obj.HeatFluxThroughWorkpiece = obj.TotalPowerBalance - obj.HeatCarriedAwayByChip - obj.HeatFluxAwayFromToolTip;
        end

        function obj = framesOverlap(obj,Frame,index)%Used when it is necessary
            %to work with different emissivities
            cTT = obj.CoordinateToolTip;
            alpha = (90 - obj.ClearanceAngle)*pi/180;
            gamma = obj.RakeAngle*pi/180;
            p1 = cTT + 67*[-cos(gamma) sin(gamma)];
            p2 = cTT + 33*[-cos(alpha) sin(alpha)];
            c = [cTT(1) p1(1) p2(1)];
            r = [cTT(2) p1(2) p2(2)];
            biTool70 = roipoly(Frame(index).e70,c,r);
            aux = biTool70 == 1 & obj.biImageChip == 1;
            biTool70 = biTool70 - aux;
            biTool70andChip = biTool70 == 1 | obj.biImageChip == 1;
            biFrame85 = ones(size(Frame(index).e85)) - biTool70andChip;
            obj.frame = biTool70andChip.*Frame(index).e70 + biFrame85.*Frame(index).e85;
        end

        function obj = toolContour(obj)%Gets a binary image of the tool
            %Provide a binary image of the tool based on the coordinates of
            %the tool tip and on the angles of the faces (rake and
            %clearance faces)
            A = round(obj.CoordinateToolTip);
            m = size(obj.frame,1);
            xt = A(1);
            yt = A(2);
            y1 = round(yt + (xt - 1)*tan(obj.RakeAngle*pi/180));
            x2 = round(xt - (m - yt)*tan(pi/2 - (90 - obj.ClearanceAngle)*pi/180));
            c = [xt 0 0 x2];
            r = [yt y1 m m];
            B = roipoly(obj.frame,c,r);
            obj.biImageTool = B;
        end

        function obj = chipContour(obj)
            %method 1-----------------------------------------------------
            %This method extracts the chip contour by taking the contour of
            %the line with 200 °C and then removes the tool contour(remaing
            %only the chip contour)
            c = obj.line200(1,:);
            r = obj.line200(2,:);
            B = roipoly(obj.frame,c,r);
            obj.biImageChip = B;
            B2 = obj.biImageTool ==1 & B == 1;
            B = B - B2;
            obj.biImageChip = B;
            imshow(B)
            K = B.*obj.frame;
            figure
            imagesc(K)
            %method 2-----------------------------------------------------
            %This method is based on colors of the temperatures
            aux = round(obj.frame);
            aux(aux < 0) = 0;
```

```
            y = label2rgb(aux,'parula');
            image(y)
            lowerlimit = 175;
            if obj.UnCutChipThickness < 400*10^-6
                lowerlimit = 165;
            end
            B1 = y(:,:,2) > lowerlimit & y(:,:,3) < 210;
            B2 = obj.biImageTool ==1 & B1 == 1;
            B = B1 - B2;
            obj.biImageChip = B;
            figure
            imshow(B)
            K = B.*obj.frame;
            figure
            imagesc(K)
            figure
            imagesc(obj.frame)
        end

        function obj = maximumTemperature(obj)
            obj.MaximumTemperatureCuttingZone = max(max(obj.frame));
            [~,lin] = max(obj.frame);
            [~,col] = max(max(obj.frame));
            lin = lin(col);
            obj.xyMaxTemp = [col lin];%Position of the maximum temperature
            %in the cutting zone
            imagesc(obj.frame)
            hold on
            plot(col,lin,'xr')
            hold off
        end

        function l = contactLength(obj)
            %Command to measure the length(in pixels) of the contact
            %between tool and chip
            imagesc(obj.frame)
            imdistline%Help to measure the amount of pixels on the contact length
            v = input('What is the value of the contact length for this frame [pixel]? ');
            close all
            l = 15*10^-6*v;
        end

        function obj = maxTemperatureTool(obj)
            %Gets the maximum temperature inside the tool contour
            C = obj.biImageTool;
            Frame = C.*obj.frame;
            T = max(max(Frame));
            obj.MaximumTemperatureTool = T;
        end

        function obj = maxTemperatureChip(obj)
            %Gets the maximum temperature inside the chip contour
            Frame = obj.biImageChip.*obj.frame;
            obj.MaximumTemperatureChip = max(max(Frame));
        end

        function obj = meanTemperatureTool(obj)
            %Get the points of valid pixels inside the tool and calculates
            %the average value for these points
            B = obj.biImageTool;
            Frame = B.*obj.frame;
            B = Frame > obj.validTemperature;
            Frame = B.*Frame;
            s = sum(sum(Frame));%Total sum of temperatures
            n = sum(sum(B));%Total number of valid pixels
            meanT = s/n;
            obj.MeanTemperatureTool = meanT;
        end

```

```
        function obj = displayBinary(obj)%Plot the binary image of the frame
            imshow(obj.BW);
            hold on
            %Plot the points that determine the CF and RF and the tool tip
            %position
            plot(obj.coordRF(:,1),obj.coordRF(:,2),'bx')
            plot(obj.coordCF(:,1),obj.coordCF(:,2),'yx')
            plot(obj.CoordinateToolTip(1),obj.CoordinateToolTip(2),'xm')
            hold off
        end

        function obj = TempTT(obj)%Calculates the temperature at tool tip
            p1 = round(obj.CoordinateToolTip + 5*[-cos(obj.RakeAngle*pi/180) sin(obj
    .RakeAngle*pi/180)]);
            p2 = round(obj.CoordinateToolTip + 5*[-cos((90 - obj.ClearanceAngle)*pi
    /180) sin((90 - obj.ClearanceAngle)*pi/180)]);
            p3 = round(obj.CoordinateToolTip + 5*[-(cos(obj.RakeAngle*pi/180)+cos
    ((90 - obj.ClearanceAngle)*pi/180)) (sin(obj.RakeAngle*pi/180)+sin((90 - obj.
    ClearanceAngle)*pi/180))]);
            T1 = obj.frame(p1(2),p1(1));
            T2 = obj.frame(p2(2),p2(1));
            T3 = obj.frame(p3(2),p3(1));
            TT = obj.frame(round(obj.CoordinateToolTip(2)),round(obj.
    CoordinateToolTip(1)));
            T = [T1 T2 T3 TT];
            obj.TemperatureToolTip = mean(T);
            %              imagesc(obj.frame)
            %              hold on
            %              plot([p1(1) p2(1) p3(1) obj.CoordinateToolTip(1)],[p1(2)
    p2(2) p3(2) obj.CoordinateToolTip(2)],'xm')
        end

        function obj = calculateCoordinates(obj)
            obj.BW = edge(obj.frame,'sobel');
            imshow(obj.BW)
            figure
            imagesc(f.*(ones(size(f))-obj.BW))
            %-------------------------Finding the clearance face-----------
            [H, THETA, RHO] = hough(obj.BW,'Theta',2:5);%Hough transformation
            %used to find angles between 2 - 5
            P  = houghpeaks(H, 10);
            obj.lines = houghlines(obj.BW, THETA, RHO, P, 'FillGap', 15,'MinLength'
    ,10);%Here we can find the lines of cutting edge and afterwards find the
    coordinate of the tool tip
            l = length(obj.lines);
            obj.coordCF = [];
            for i=1:l%Loop to test the resultant lines from the command houghlines
                Theta=obj.lines(i).theta;
                t1 = obj.lines(i).point1;
                t2 = obj.lines(i).point2;
                rho = obj.lines(i).rho;
                imagesc(f)
                hold on
                plot([t1(1) t2(1)]',[t1(2) t2(2)]','m')
                hold off
                if rho < 204 && rho > 198
                    obj.coordCF = [t1;t2];
                    obj.ClearanceAngle = Theta;
                end
            end
            %-------------------------Finding the rake face----------------
            [H, THETA, RHO] = hough(obj.BW,'Theta',81:85);%Hough transformation
            %used to find angles between 5 - 9 (must be the complementary on this
    direction)
            P  = houghpeaks(H, 10);
            obj.lines = houghlines(obj.BW, THETA, RHO, P, 'FillGap', 15,'MinLength'
    ,10);%Here we can find the lines of cutting edge and afterwards find the
    coordinate of the tool tip
            l = length(obj.lines);
            obj.coordRF = [];
```

```
            for i=1:l
                Theta=obj.lines(i).theta;
                t1 = obj.lines(i).point1;
                t2 = obj.lines(i).point2;
                rho = obj.lines(i).rho;
                imagesc(f)
                hold on
                plot([t1(1) t2(1)]',[t1(2) t2(2)]','m')
                hold off
                if rho < 103 && rho > 98
                    obj.coordRF = [t1;t2];
                    obj.RakeAngle = 90 - Theta;
                end
            end
        end

        function obj = coordinateToolTip(obj)
            a = (obj.coordRF(1,2) - obj.coordRF(2,2))/(obj.coordRF(1,1) - obj.coordRF(2,1));%The slope of the rake face hardly will be Inf(Infinite) or NaN(Not-a-number),
            %because we took for this face a slope smaller than 45?
            b = obj.coordRF(1,2) - a*obj.coordRF(1,1);
            m = (obj.coordCF(1,2) - obj.coordCF(2,2))/(obj.coordCF(1,1) - obj.coordCF(2,1));%Slope of the cf, in some cases may be Inf(inclination of 90?, for example)
            h = @(x)(a*x + b);%line of the clearance face represented by f
            if m == Inf || m == -Inf%if the slope of the cf is 90? or -90?(Inf or -Inf)
                xi = obj.coordCF(1,1);%xi represents the coordinate x of the intersection(tool tip)
            else
                n = obj.coordCF(1,2) - m*obj.coordCF(1,1);
                xi = (n - b)/(a - m);
            end
            yi = h(xi);
            obj.CoordinateToolTip = [xi yi];
        end

        function obj = displayImageAndToolTip(obj)
            %Plots the fram image and the tool tip point
            figure
            imagesc(obj.frame);
            hold on
            plot(obj.CoordinateToolTip(1),obj.CoordinateToolTip(2),'xm')
            hold off
        end

        function obj = pointsRFandCF(obj)
            %This function gets points to help to plot temperature behavior
            %along the rake, clearance face and the bissectrix between these
            %faces
            alpha = (90 - obj.ClearanceAngle)*pi/180;
            gamma = obj.RakeAngle*pi/180;
            obj.pointRF = obj.CoordinateToolTip + 90*[-cos(gamma) sin(gamma)];
            obj.pointCF = obj.CoordinateToolTip + 90*[-cos(alpha) sin(alpha)];
            obj.pointM = obj.CoordinateToolTip + 40*[-2*cos(alpha)-cos(gamma) 2*sin(alpha)+sin(gamma)];
        end

        function vT = temperatureRFandCF(obj)
            pixelpitch = 15*10^-3;% mm/pixel
            extCF = obj.pointCF;% final point on the clearance face
            extRF = obj.pointRF;% final point on the rake face
            extM = obj.pointM;%final point of the bissectrix
            l1 = round(abs(obj.CoordinateToolTip(1)-extRF(1)));%length in pixels of rake line
            l2 = round(abs(obj.CoordinateToolTip(2)-extCF(2)));%length in pixels of clearance line
            l3 = max(round(abs(obj.CoordinateToolTip-extM)));
```

```
            vRFx = round(linspace(obj.CoordinateToolTip(1),extRF(1),l1));%coordinates x of the rake line
            vRFy = round(linspace(obj.CoordinateToolTip(2),extRF(2),l1));%coordinates y of the rake line
            vCFx = round(linspace(obj.CoordinateToolTip(1),extCF(1),l2));%coordinates x of the clearance line
            vCFy = round(linspace(obj.CoordinateToolTip(2),extCF(2),l2));%coordinates y of the clearance line
            vMx = round(linspace(obj.CoordinateToolTip(1),extM(1),l3));
            vMy = round(linspace(obj.CoordinateToolTip(2),extM(2),l3));
            T_RF = zeros(1,l1);%temperature for each pixel (each coordinate pair) - rake line
            T_CF = zeros(1,l2);%temperature for each pixel (each coordinate pair) - clearance line
            T_M = zeros(1,l3);
            for t=1:l1
                T_RF(t) = obj.frame(vRFy(t),vRFx(t));%Building the temperature vector - rake line
            end
            for t=1:l2
                T_CF(t) = obj.frame(vCFy(t),vCFx(t));%Building the temperature vector - clearance line
            end
            for t=1:l3
                T_M(t) = obj.frame(vMy(t),vMx(t));%Building the temperature vector - clearance line
            end
            d1 = zeros(1,l1);%distance for each pixel along the line
            d2 = zeros(1,l2);
            d3 = zeros(1,l3);
            for t=1:l1 - 1
                d1(t+1)=(((vRFx(t+1)-vRFx(1))^2)+((vRFy(t+1)-vRFy(1))^2))^(1/2);
            end
            for t=1:l2 - 1
                d2(t+1)=(((vCFx(t+1)-vCFx(1))^2)+((vCFy(t+1)-vCFy(1))^2))^(1/2);
            end
            for t=1:l3 - 1
                d3(t+1)=(((vMx(t+1)-vMx(1))^2)+((vMy(t+1)-vMy(1))^2))^(1/2);
            end
            d1 = d1*pixelpitch;
            d2 = d2*pixelpitch;
            d3 = d3*pixelpitch;
            figure
            hold on
            plot(d1,T_RF)
            plot(d2,T_CF)
            plot(d3,T_M)
            xlabel('Distance from the tool tip (mm)')
            ylabel('Temperature (°C)')
            legend('Rake face','Clearance face','Middle vector')
            %saveas(fig,file,'jpeg')
            hold off
            figure
            imagesc(obj.frame)
            colormap jet
            hold on
            plot(vRFx,vRFy,'k','LineWidth',1)
            plot(vCFx,vCFy,'k','LineWidth',1)
            plot(vMx,vMy,'k','LineWidth',1)
            hold off
            m = min([l1 l2 l3]);
            vT = [d1(1:m)' T_RF(1:m)' d2(1:m)' T_CF(1:m)' d3(1:m)' T_M(1:m)'];
            %Not only plot the graph but also export the points of
            %Temperature x distance
        end

        function obj = extremePointsChip(obj)
            [y,x] = find(obj.lineChip);%gets a range of the coordinates of this line
            %The first point is on the highest position and the last is on
            %the lowest(which are the extremity of the line)
```

```
            obj.extPtosLineChip = [x(1) y(1);x(end) y(end)];
        end

        function obj = displayIsotherms(obj)
            tRF = obj.RakeAngle*pi/180;
            tCF = (90 - obj.ClearanceAngle)*pi/180;
            vRF = [-cos(tRF) sin(tRF)];
            vCF = [-cos(tCF) sin(tCF)];
            %p1 RF direction
            t = (obj.CoordinateToolTip(1) - 1)/vRF(1);
            p1 = obj.CoordinateToolTip - t*vRF;
            %p2 CF direction
            t = (256 - obj.CoordinateToolTip(2))/vCF(2);
            p2 = obj.CoordinateToolTip + t*vCF;
            %auxiliar to plot
            auxX = [p1(1) obj.CoordinateToolTip(1) p2(1)]';
            auxY = [p1(2) obj.CoordinateToolTip(2) p2(2)]';
            Tmax = max(max(obj.biImageTool.*obj.frame));
            Tv = obj.validTemperature;
            v = round(Tv:40:Tmax);
            %Display only isotherms and tool contour-----------------------
            figure
            [C,h] = contour(obj.frame,v);
            hold on
            clabel(C,h,'FontSize',10)
            h.LineWidth = 1.5;
            daspect([1,1,1])
            plot(auxX,auxY,'k')
            text(150,150,'Tool')
            text(100,75,'Chip')
            xlabel('pixel');
            ylabel('pixel');
            title('Isotherms');
            cb = colorbar('vert');
            zlab = get(cb,'ylabel');
            set(zlab,'String','Temperature (°C)');
            axis([50 250 25 200])
            %saveas(fig,t,'jpeg')
            axis off
            colormap jet
            hold off
            %Display tool and isotherms-----------------------------------
            lc = obj.extPtosLineChip;
            figure
            imagesc(obj.frame)
            colormap jet
            hold on
            plot(auxX,auxY,'k')
            plot(lc(:,1),lc(:,2),'k--','LineWidth',1)%line in the chip on
            %the end of contact
            [C,h] = contour(obj.frame,v);
            h.LineColor = [0.247 0.247 0.247];%Set only one color for the
            %contour lines
            clabel(C,h,'manual','FontSize',10);
            x = obj.CoordinateToolTip(1);
            y = obj.CoordinateToolTip(2);
            axis([x-180 x+15 y-60 y+110])%resize the image
            daspect([1,1,1])
            axis off
            hold off
        end

        function obj = calculateGradient(obj)
            pp = 15*10^-6;%pixelpitch in m
            tx = zeros(size(obj.frame));
            ty = zeros(size(obj.frame));
            k = 0;
            for j = 1:5%loop to calculate the gradients of temperature
                [auxx,auxy]=gradaux_v2(obj.frame,j);
                tx = tx + auxx;
```

```
                ty = ty + auxy;
                k = k + 1;
            end
            obj.Tx = tx/(k*pp);
            obj.Ty = ty/(k*pp);
        end

        function obj = displayGradient(obj)
            %Display the gradient arrows for tool tip are, part of the rake
            %face and clearance face
            auxx = [obj.pointCF(1) obj.CoordinateToolTip(1) obj.pointRF(1)];
            auxy = [obj.pointCF(2) obj.CoordinateToolTip(2) obj.pointRF(2)];
            k = obj.heatConductivity;
            qx = -k*obj.Tx;
            qy = -k*obj.Ty;
            figure
            quiver(qx,qy)
            hold on
            plot(auxx,auxy,'k')
            xmin = obj.CoordinateToolTip(1) - 10;
            xmax = obj.CoordinateToolTip(1) + 5;
            ymin = obj.CoordinateToolTip(2) - 5;
            ymax = obj.CoordinateToolTip(2) + 10;
            axis([xmin xmax ymin ymax])
            title('Tool Tip')
            daspect([1,1,1])
            figure
            quiver(qx,qy)
            hold on
            plot(auxx,auxy,'k')
            xmin = obj.CoordinateToolTip(1) - 30;
            xmax = obj.CoordinateToolTip(1) - 10;
            ymin = obj.CoordinateToolTip(2) - 5;
            ymax = obj.CoordinateToolTip(2) + 15;
            axis([xmin xmax ymin ymax])
            title('Rake Face')
            daspect([1,1,1])
            figure
            quiver(qx,qy)
            hold on
            plot(auxx,auxy,'k')
            xmin = obj.CoordinateToolTip(1) - 10;
            xmax = obj.CoordinateToolTip(1) + 10;
            ymin = obj.CoordinateToolTip(2) + 10;
            ymax = obj.CoordinateToolTip(2) + 20;
            axis([xmin xmax ymin ymax])
            title('Clearance Face')
            daspect([1,1,1])
        end

        function obj = displayGradientContour(obj)
            auxx = [obj.pointCF(1) obj.CoordinateToolTip(1) obj.pointRF(1)];
            auxy = [obj.pointCF(2) obj.CoordinateToolTip(2) obj.pointRF(2)];
            k = obj.heatConductivity;
            qx = -k*obj.Tx;
            qy = -k*obj.Ty;
            figure
            quiver(qx,qy)
            hold on
            plot(auxx,auxy,'k')
            contour(obj.frame,10)
            xmin = obj.CoordinateToolTip(1) - 20;
            xmax = obj.CoordinateToolTip(1) + 5;
            ymin = obj.CoordinateToolTip(2) - 5;
            ymax = obj.CoordinateToolTip(2) + 20;
            axis([xmin xmax ymin ymax])
            daspect([1,1,1])
        end

        function obj = findLineChip(obj)%Gets the line perpendicular to the
```

```
            %chip movement on the end of the contact tool - chip
            [m,n] = size(obj.frame);
            o = obj.RakeAngle*pi/180;
            l = obj.ContactLength/(15*10^-6);
            c = obj.CoordinateToolTip + l*[-cos(o) sin(o)];
            xm = c(1);
            ym = c(2);
            x1 = xm - tan(o)*(ym - 1);
            x2 = x1 + tan(o)*(m - 1);
            vx = round(linspace(x1,x2,m));
            vy = linspace(1,m,m);
            B1 = zeros(m,n);
            for i = 1:m
                B1(vy(i),vx(i)) = 1;
            end
            B2 = B1 == 1 & obj.biImageChip == 1;
            obj.lineChip = B2;
            imshow(B2)
            figure
            imshow(obj.biImageChip)
            aux = (obj.biImageChip - B2).*obj.frame;
            figure
            imagesc(aux)
        end

        function obj = findLineTool(obj)%Function that gets the coordinates of
            %the contour lines for each temperature level
            [m,n] = size(obj.frame);
            Tmax = max(max(obj.biImageTool.*obj.frame));
            Tv = obj.validTemperature;
            v = round(Tv:40:Tmax);%the step between the temperatures is 40
            if length(v) == 1%Conditional if the min and max temperatures are equal
                v = round([Tv Tmax]);
            end
            [C,~] = contour(obj.frame,v);%Getting the contour lines for C
            close
            l = length(v);
            B = zeros(m,n,l);
            C = round(C);
            for k = 1:l
                [~,J] = find(C == v(k));
                [~,p] = max(C(2,J));
                J = J(p);
                for z = J+1:J+C(2,J)
                    B(C(2,z),C(1,z),k) = 1;
                end
                if k == 1%Take the last line for the future heat flow calculation
                    obj.line200 = C(:,J+1:J+C(2,J));
                end
                B(:,:,k) = B(:,:,k).*obj.biImageTool;
            end
            obj.lineTool = B;
            for i = 1:l %Loop to check the binary image of the lines
                imshow(B(:,:,i))
            end
        end

        function obj = heatBalance(obj,tuc,Vc,ap,w)
            pp = 15*10^-6;% pixel pitch [m/pixel]
            %
    -------------------------------------------------------------------------
            %First part - Heat carried away by the chip
            k = obj.heatConductivity;% Heat conductivity tool
            cp = obj.heatCapacityWP;
            M = obj.lineChip.*obj.frame;
            MH = polyval(cp,M);
            MH(MH == cp(end)) = 0;%Remove the values for tempertures equal to zero
            Ht = MH.*(obj.frame-22);%J/kg - 22 is the temperature of the environment
            Ht = sum(sum(Ht));
            n = sum(sum(obj.lineChip));
```

```
            Hc = Ht/n; %mean entalpy on the line chip
            p = 7697; %density of workpiece - kg/m^3
            Qc = Hc*Vc*tuc*p;%Vc*tuc is equals to Vchip*tchip
            obj.HeatCarriedAwayByChip = Qc*ap;
            %-------------------------------------------------------------------------
            %Second part - Heat carried away by the tool
            dT = ((obj.Tx).^2 + (obj.Ty).^2).^(1/2);%Resultant gradient of temperature
            Q = zeros(size(obj.lineTool,3),1);
            for i = 1:size(obj.lineTool,3)
                L = obj.lineTool(:,:,i);
                Q(i) = sum(sum(L.*dT))*pp*k*w;
            end
            obj.heatAccumulatedPerLine = Q;
            Qm = mean(Q(1:2));%mean of the last two lines
            obj.HeatFluxAwayFromToolTip = Qm;
        end

        function n = exceedingPoints(obj,Temperature)
            B = obj.frame.*obj.biImageTool > Temperature;
            n = sum(sum(B));
%             K = sprintf('Number of points exceeding the defined temperature is: %3d \n',n);
%             disp(K)
        end

        function obj = internalEnergyTool(obj,w)%Calculates the internal energy
            %in the tool, but only for valid pixels
            pp = 15*10^-6;%pixel pitch in cm
            cp = obj.heatCapacityTool;%J/(K*m^3)
            %Heat capacity for tool
            Te = 22;%Environment
            B = obj.frame.*obj.biImageTool > obj.validTemperature;
            B1 = obj.frame.*B;
            B2 = polyval(cp,B1);%Heat capacity for each pixel (J/kgK)
            B2(B2 == cp(4)) = 0;
            H = B2.*(obj.frame - Te)*(pp^2);%Heat Amount for each pixel(J/m)
            Ha = sum(sum(H));%Mean value for the entire tool
            obj.InternalEnergyTool = Ha*w;%w is the width of the workpiece
        end

        function obj = shearLine(obj)%This function tries to identify the
            %of the primary shear zone (to get the shear angle)
            B = obj.biImageChip;
            v1 = sum(B);
            v1(v1 == 0) = [];
            l1 = length(v1);
            C = imcrop(B,[20 20 l1 100]);%Cutting the image to a new size
            %to focus on the shear region
            [m,n] = size(C);
            pto = zeros(1000,2);
            count = 1;
            %In the following loop we create a binary image of the chip
            %contour without its boundaries which is necessary to identify the shear line
            for i = 2:m-1
                for j = 2:n-1
                    if C(i,j+1) == 1 && C(i,j-1) == 1 && C(i+1,j) == 1 && C(i-1,j) == 1
                        pto(count,:) = [i j];
                        count = count + 1;
                    end
                end
            end
            %On the following loop the empty cells are removed
            for i =1000:-1:1
                if isequal(pto(i,:),[0 0]) == 1
                    pto(i,:) = [];
                end
```

```
            end
            l = size(pto,1);
            %In this loop only the boundaries of the chip is left
            for i = 1:l
                C(pto(i,1),pto(i,2)) = 0;
            end

            [H, THETA, RHO] = hough(C,'Theta',-40:-30);%Hough transformation
            %from 30 until 40 is the range of possible angles for the shear
            %angle
            P  = houghpeaks(H, 5);
            lin = houghlines(C, THETA, RHO, P, 'FillGap', 15,'MinLength',10);
            l=length(lin);
            %p1 and p2 will be the extreme points of the shear line
            p1 = [];
            p2 = [];
            for i=1:l
                Theta=lin(i).theta;
                t1 = lin(i).point1;
                t2 = lin(i).point2;
                y = abs(t1(2)-t2(2));
%                 imshow(C)
%                 hold on
%                 plot([t1(1) t2(1)]',[t1(2) t2(2)]','xm')
%                 hold off
                if isempty(p1) && isempty(p2) && abs(Theta + 34) < 5
                    p1 = t1 + [19 19];%Plus 19 due to the previous cutting image process
                    p2 = t2 + [19 19];
                    ym = y;
                    obj.ShearAngle = abs(Theta);
                end
                if abs(Theta + 34) < 5 && y > ym
                    p1 = t1 + [19 19];
                    p2 = t2 + [19 19];
                    obj.ShearAngle = abs(Theta);
                end
            end
            if isempty(obj.ShearAngle)%It was not possible to find a good line
                obj.ShearAngle = 30;%Default condition
            end
        end

        function obj = forcesValues(obj,Fp,Fq,ap,tuc)
            phi = obj.ShearAngle*pi/180;%shear angle
            gamma = obj.RakeAngle*pi/180;%Rake angle
            Fs = Fp*cos(phi) - Fq*sin(phi);%Cutting force component parallel to shear plane
            Ns = Fq*cos(phi) + Fp*sin(phi);%Cutting force component perpendicular to shear plane
            Fc = Fp*sin(gamma) + Fq*cos(gamma);%Cutting force component parallel to tool face
            Nc = Fp*cos(gamma) - Fq*sin(gamma);%Cutting force component perpendicular to tool face
            mu = Fc/Nc; % coefficient of friction
            As = ap*tuc/sin(phi);%Area shear plane
            tau = Fs/As;%shear stress
            sigma = Ns/As;%Normal stress
            r = sin(phi)/cos(phi - gamma); %ratio r = t/tc = lc/l
            ss = cos(gamma)/(sin(phi)*cos(phi-gamma));%shear strain
            us = tau*ss;%shear energy per volume
            uf = Fc*r/(tuc*ap);%friction energy per volume
            beta = atan(Fc/Nc);%friction angle on tool face
            obj.CuttingForceParallelToolFace = Fc;
            obj.CuttingForcePowerDirection = Fp;
            obj.CuttingForceUncutChipThicknessDirection = Fq;
            obj.CuttingForceParallelShearPlane = Fs;
            obj.CuttingForcePerpendicularShearPlane = Ns;
            obj.CuttingForcePerpendicularToolFace = Nc;
            obj.CoefficientFriction = mu;
```

```
            obj.ShearStress = tau;
            obj.NormalStress = sigma;
            obj.RatioR = r;
            obj.ShearEnergyVolume = us;
            obj.FrictionEnergyVolume = uf;
            obj.FrictionAngle = beta*180/pi;
        end

        function obj = calculatePecletNumber(obj)
            cp = polyval(obj.heatCapacityWP,obj.MaximumTemperatureCuttingZone);
            %heat capacity for the maximum temperature(J/kgK)
            k = 39.1;%heat conductivity of the workpiece (W/mK) - mean value
            d = 7.697*10^3;%density of the tool (kg/m^3)
            obj.PecletNumber = ((obj.CuttingVelocity/60)*obj.UnCutChipThickness)/(k
    /(cp*d));
        end
    end
end
```

FOLHA DE REGISTRO DO DOCUMENTO

1. CLASSIFICAÇÃO/TIPO	2. DATA	3. DOCUMENTO Nº	4. Nº DE PÁGINAS
TC	20 de novembro de 2017	DCTA/ITA/TC-031/2017	64

5. TÍTULO E SUBTÍTULO:
Computational method for temperatures and heat flows analysis of orthogonal cutting 1045 steel by thermal imaging

6. AUTORA(ES):
Adriana Nunes Chaves Lima

7. INSTITUIÇÃO(ÕES)/ÓRGÃO(S) INTERNO(S)/DIVISÃO(ÕES):
Instituto Tecnológico de Aeronáutica – ITA/IEM

8. PALAVRAS-CHAVE SUGERIDAS PELA AUTORA:
Manufacture; Image Processing; Matlab; Thermal Analysis; Orthogonal Cutting; Software.

9. PALAVRAS-CHAVE RESULTANTES DE INDEXAÇÃO:
Fabricação; Processamento de Imagens; Programas; Análise Térmica; Engenharia mecânica.

10. APRESENTAÇÃO: (**X**) **Nacional** ( ) **Internacional**
ITA, São José dos Campos. Curso de Graduação em Engenharia Mecânica. Orientador: Anderson Vicente Borille; coorientador: Thorsten Augspurger. Publicado em 2017.

11. RESUMO:
Methods for inspection and monitoring have been used more and more to ensure the quality of processes. In the machining field there are many important process parameters to assure that the expected results are achieved. The surface quality of a workpiece and the tool life, for example, are directly influenced by the thermal energy generated in the heat zones. Due to it, there are lots of theoretical methods for temperature modeling along the cutting zone, but there is still a lack of tools able to allow for practical validation of these methods. Although many challenges still prevail for the adequate use of thermography, this technology enables the development of computational methods for processing of thermal images and, consequently, the heat flow and heat partition analysis. This paper comes to present a computational method developed on MATLAB with the support of image processing toolbox. It performs thermal image analysis, providing results regarding temperature fields, inner energies, heat flows and other variables of interest that can be used on machining monitoring and future studies to improve cutting parameters.

12. GRAU DE SIGILO:
(**X**) **OSTENSIVO** ( ) **RESERVADO** ( ) **SECRETO**