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.
54f.

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. 54f. 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

I would like to thank my family for everything they have done and still do for me, with endless love and comprehension. They are the main reason of this accomplishment, telling me always the importance of education in my life, which comes to be the best present they could give to me. My parents Ivanildo and Conceição, who demonstrated huge commitment with hard work, intending to provide better schools for me and my siblings. I also especially thank my siblings Aline and Alisson, who were my great examples of perseverance and the first ones to encourage me to pursue my dream to become an engineer from ITA.

I thank my roommates over my time at H8, Jéssica Meireles, Giuliana Warda, Sarah Borges, Eduarda Campos and Dafne de Brito. Also, I would like to thank my dear friends Camilla Matias, Nathianne de Moura and Raffaela Führ. Thank you all for helping me to grow up and relieve my longing for home.

I thank Thorsten Augspurguer, who provided me guidance during my internship in Germany, which was a great professional and personal experience. I also thank my coworkers at the time and friends Florisa Gessle and Renata Lie for their support and for having made my days even better.

Finally, I thank Abraão Barros for the motivation you provided me. Every opportunity you made me see, the strength you gave me when I needed most and your support to my professional aims were essential to make my path a lot easier. Thank you for being part of my life.

"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 each time more to guarantee the quality of processes. In the machining field there are many important parameters to assure that a process arranges the designed results. The superficial finishing of a workpiece and the tool life are some examples subjected to the direct influence of thermal energy generated in the heat zones. Due to it, there are a lot of theoretical methods for temperature modeling along the cutting zone, but still there is a lack of ways able to allow practical validation of these methods. Although many challenges still prevail on the suitable use of thermography, this technology makes possible 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 image processing toolbox support. It makes thermal image analysis, providing results about 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

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

Contents

1 Introduction

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 that orthogonal cutting conditions can be observed.

Also, it is known that thermal behavior during cutting processes, as temperature fields and heat flows, has an important influence on tool life, surface finish and metallurgical structure of workpiece and machinability. Then, the study of thermal analysis on orthogonal cutting case shall be able to provide a better comprehension of many studies concerning thermal modeling of metal cutting (KOMANDURI; HOU, 2000), (KOMANDURI; HOU, 2001).

1.1 Overview of metal cutting

There are different ways to modify raw material, as additive and subtractive (SHAW; COOKSON, 2005). The additive processes occur when separated materials are put together, like 3D printing or welding. On the other hand, the subtractive way removes unnecessary material, which happens for machining processes as turning, milling and, in this paper, orthogonal cutting. The cutting process is composed basically by chip, tool and workpiece (figure 2.1). Many parameters are responsible for a good performance and final result, as surface finish 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 damage the expected result and the process itself.

1.2 Objective

The aim of this paper is to develop a computational method to analyze thermal images generated during orthogonal cutting of AISI 1045 metal, focusing on the transient state due to the short time of cutting. It will be analyzed temperature distribution along the cutting tool, heat flows through tool, chip and workpiece.

1.3 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 conducted 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 Thermodynamics

2.1.2 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 plane of contact between tool and chip, also known as secondary shear zone or friction zone. As for third one, it is related to the wear caused due the friction between tool and finished workpiece surface, due to it is called wear zone.

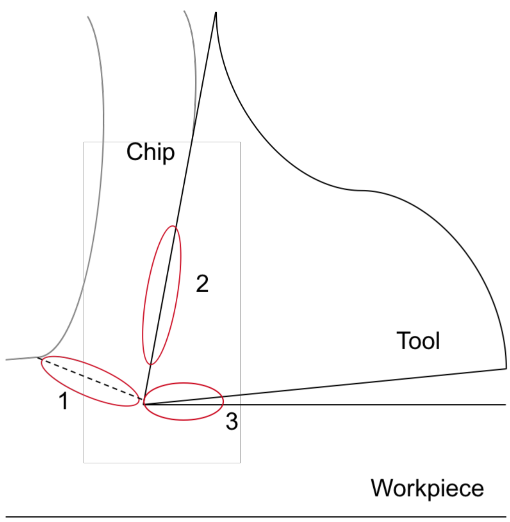

FIGURE 2.1 – Regions of interest

2.1.3 Infrared thermography operation

Infrared termography is a non-contact way to measure infrared electromagnetic energy. The humam eye can not detect the range of infrared radiation. However, there are infrared cameras able to detect this energy and process the radiation into information (figure 2.2).

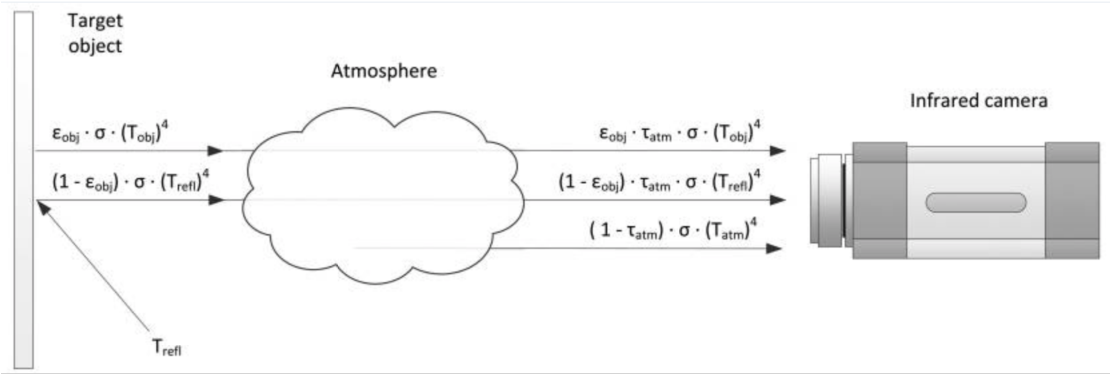

FIGURE 2.2 – Radiation received by infrared camera (USAMENTIAGA *et al.*, 2014)

It makes possible all thermal energy produced during a cutting process to be received by the infrared camera and to be synthesize in a matrix of temperatures afterwards. 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 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 way occurs when the subject matter has its temperature different from the environment (often higher). On the other hand, the active way 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 on the target's temperature measurement. To correct the situation, the IR camera has a 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 tempareture 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 numerous 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).

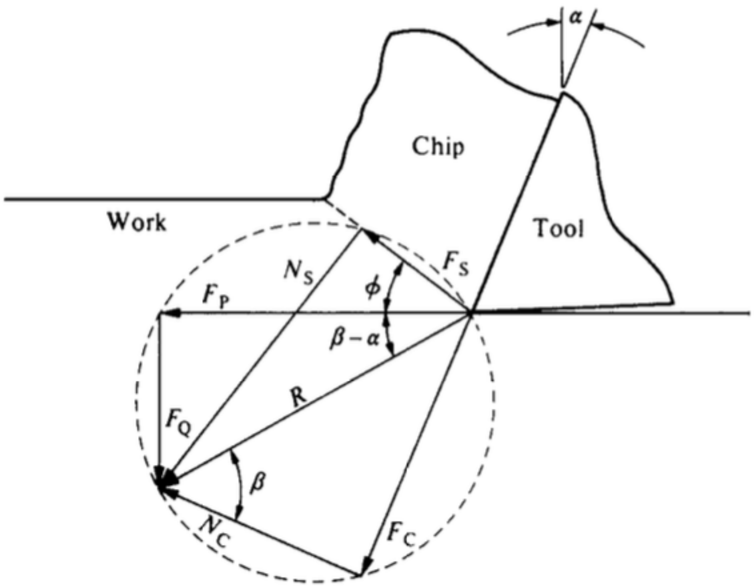

FIGURE 2.3 – Cutting forces (SHAW; COOKSON, 2005)

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

$$F_S = F_P \cos\phi - F_Q \sin\phi \tag{2.1}$$

$$N_S = F_Q \cos\phi + F_P \sin\phi \tag{2.2}$$

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

$$F_C = F_P \sin\alpha + F_Q \cos\alpha \tag{2.3}$$

$$N_C = F_P \cos\alpha - F_Q \sin\alpha \tag{2.4}$$

These equations provide all auxiliary forces related to the known passive force F_Q and force on the cutting direction F_P. Now the variables of interest can be easily calculated, as the friction coefficient:

$$\mu = \frac{F_C}{N_C} = \frac{F_Q + F_P \tan\alpha}{F_P - F_Q \tan\alpha} \tag{2.5}$$

Now the equations concerning about stresses are:

$$A_S = \frac{wa_p}{\sin\phi} \tag{2.6}$$

$$\tau = \frac{F_S}{A_S} = \frac{(F_P \cos\phi - F_Q \sin\phi)\sin\phi}{wa_p} \tag{2.7}$$

$$\sigma = \frac{N_S}{A_S} = \frac{(F_P \sin\phi + F_Q \cos\phi)\sin\phi}{wa_p} \tag{2.8}$$

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 is found experimentally that there is no change in density of metal during the cutting process and also when $w/a_p \geq 5$ makes the width of the chip the same of the workpiece. Then, the equations are:

$$a_p w l = a_{pc} w_c l_c \tag{2.9}$$

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:

$$r = \frac{a_p}{a_{pc}} = \frac{l_c}{l} \tag{2.10}$$

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

$$v_c = rv \tag{2.11}$$

2.3 State of the Art

2.3.1 Infrared Termography

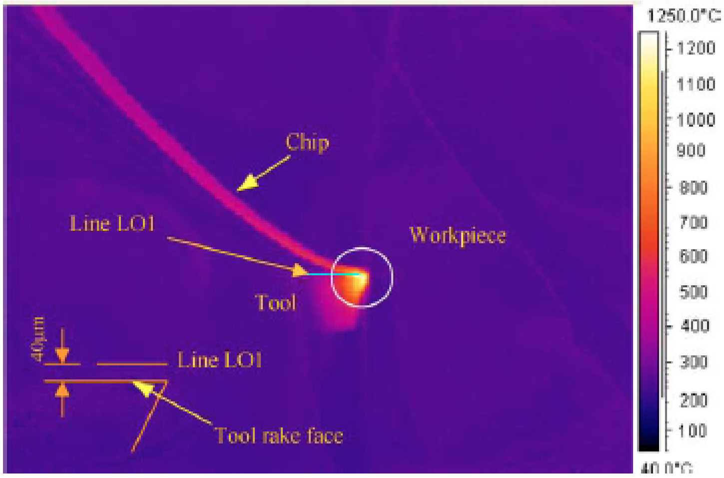

FIGURE 2.4 – Infrared photography of a cutting process (ABUKHSHIM *et al.*, 2006)

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)
- Determine a suitable emissivity is a chalenge (it changes with temperature variation)

2.3.2 Image Processing

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

Colorful 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.

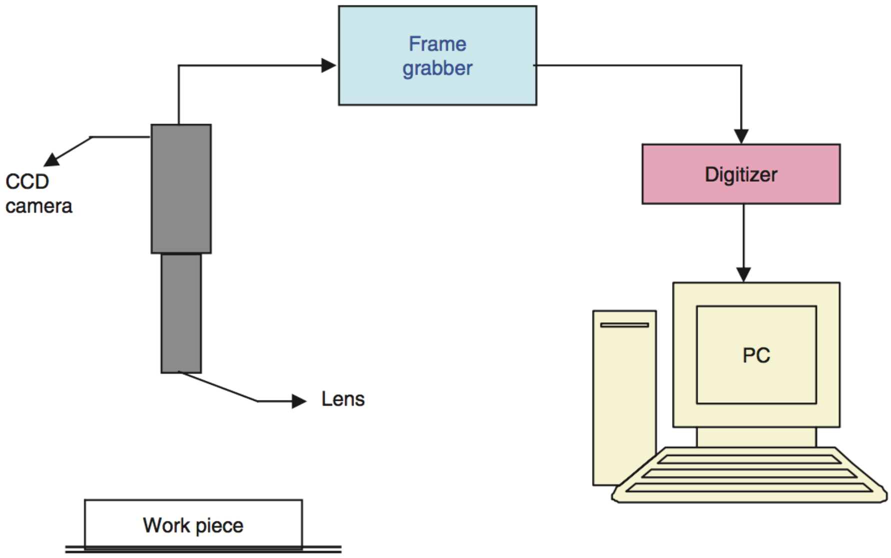

FIGURE 2.5 – Diagram of machine vision system (SARMA *et al.*, 2009)

There are many applications of image processing for 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 process, 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 few examples of what image processing can do for machining industry. There are uncountable other ways 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 systems of vision to be incorporated in online monitoring and then providing a real time feedback.

3 Materials and Methods

3.1 Experimental Setup and Materials

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

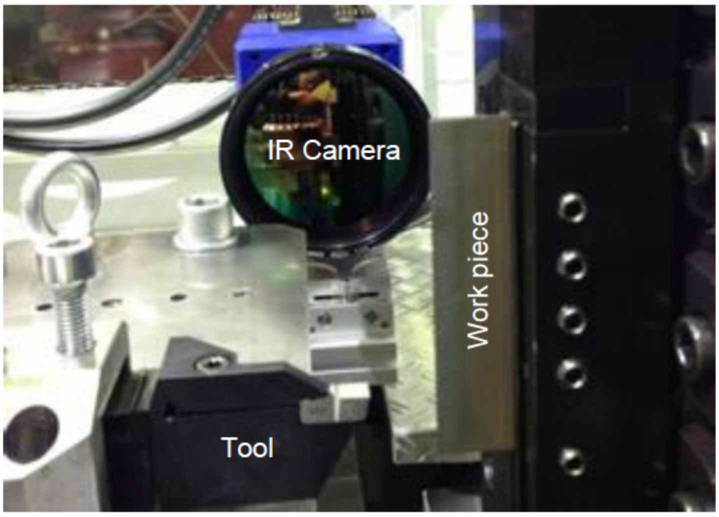

FIGURE 3.1 – Experimental setup (AUGSPURGER *et al.*, 2016)

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

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

For force acquisition during the process it was used a three-component piezoelectric force platform, 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 it was taken a mean value to be used on power calculation, equation 3.3.

All the experiments were held without coolant, with cutting velocities 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 to export 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.*, 2016)

As a machining process, the orthogonal cutting performance is subjected to many parameter like material of workpiece, 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 of real conditions. Then, all the input data necessary can be summarize 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.*, 2016)

The heat capacities of tool and workpiece material 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.*, 2016)

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.*, 2016)

3.2 Methods

3.2.1 Power calculation

This is a simple method that is stated on the following 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).

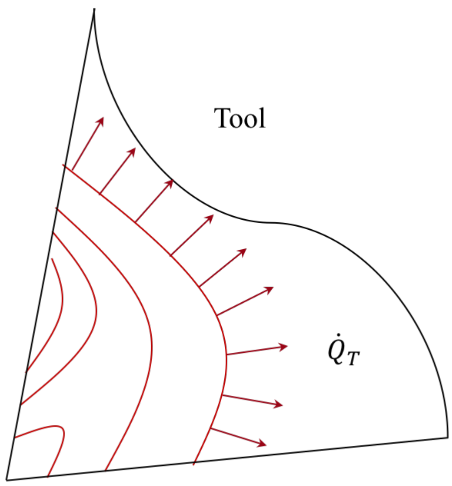

FIGURE 3.2 – Heat flow through tool

Besides the temperature matrix to calculate heat flow through tool it is necessary 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 chip normal to the line of end of contact v_{chip}, the chip thickness t_C and the chip width w.

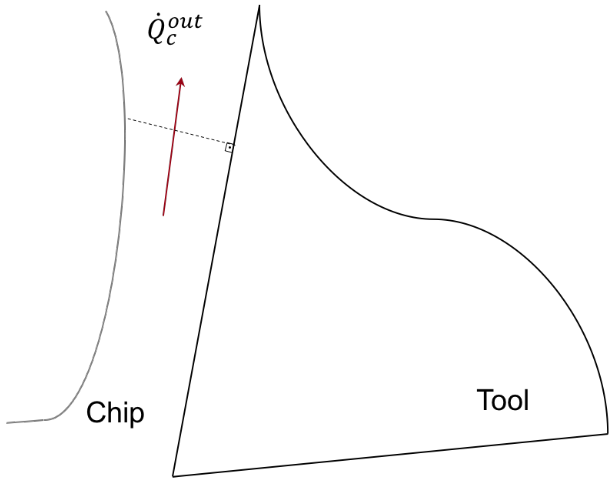

FIGURE 3.3 – Thermal energy carried away by chip

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 contact chip - tool, the math necessary to perform these equations is simple, providing reliable outcomes.

3.2.3 Volume control

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

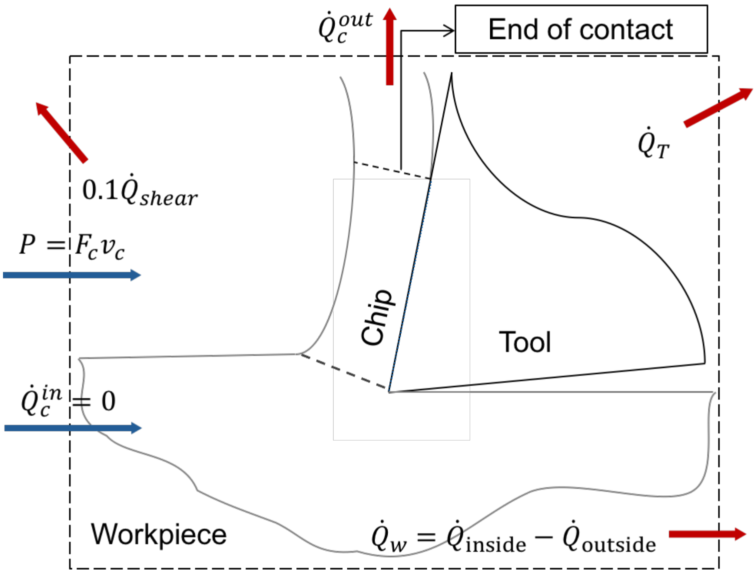

FIGURE 3.4 – Control volume

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, 1942), the others 10% are soon dissipated out the control volume. Then, 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 as output variables, which are MATLAB format of variables. Each pixel contains temperature information about itself, it is possible to visualize an example on a scaled image on the following figure:

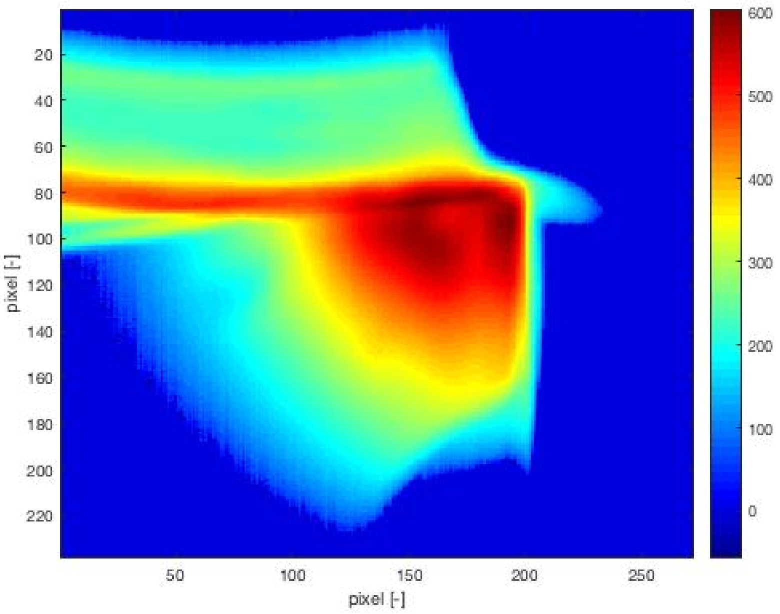

FIGURE 4.1 – Scaled image showing temperature distribution

From figure 4.1 with MATLAB Image Processing Toolbox support it possible to extract some informations about the image, such as:

- Edges recognition

- Image segmentation for tool, chip and workpiece
- Detection of tool tip
- Determine isotherms along tool

4.1.2 Auxiliar functions

4.1.2.1 Contour plot

This is an important tool for this paper, contour plot is able to provide same level curves. Since the variable used on the process is a temperature matrix, this tool will calculate continuous lines, which the temperature of each pixel has very close value. Doing it with a small tolerance, the lines calculated are isotherms of the image. Then, with these lines it is also possible to extract its coordinates, which it will be essential to calculate heat carried away from volume control by means of tool.

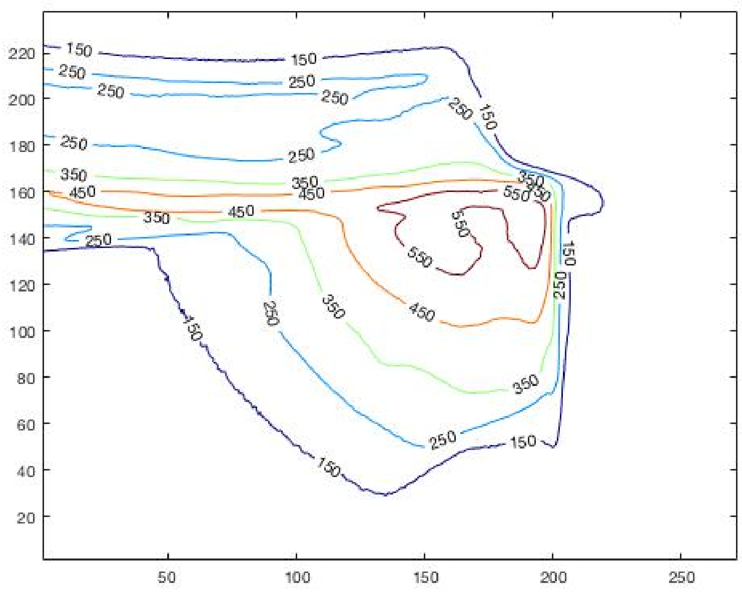

FIGURE 4.2 – Contour plot

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). Concerning about 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 what the angular coefficient of the sought lines are defined. More precise is this range, more reliable and faster will be the output.

The test bench, where the experiments were held, allows a fixed placement of tool.

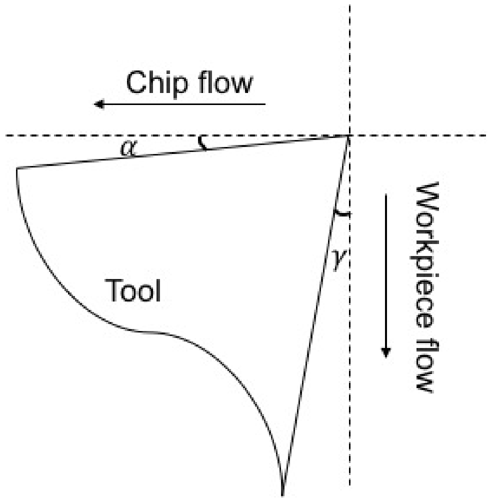

FIGURE 4.3 – Placement of tool

It means 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, being very accurate. As the rake and clearance angle are always 6^o and 3^o, respectively, the hough transform processing will last a shorter time with predetermined angles than otherwise.

4.1.3 Implementation steps

4.1.3.1 Overview

The method of the program was able to identify the tool and chip shapes, then the analysis could extract and provide features that were essential for the results of this paper. By means of image processing and some input data, features like maximum cutting zone temperature, maximum chip temperature, heat flows 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 method to find tool edges has to provide an accurate range of angles that the rake and clearance angles are inserted. The process is simple and it is demonstrated as follows:

```
1     function obj = calculateCoordinates(obj)
2         obj.BW = edge(obj.frame,'sobel');
3     %-------------------Finding the clearance face----------------------
4         [H, THETA, RHO] = hough(obj.BW,'Theta',2:5);%Hough transformation
5         P  = houghpeaks(H, 10);
6         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
7         l = length(obj.lines);
8         obj.coordCF = [];
9         for i=1:l
10            Theta = obj.lines(i).theta;
11            t1 = obj.lines(i).point1;
12            t2 = obj.lines(i).point2;
13            rho = obj.lines(i).rho;
14            if rho < 204 && rho > 198
15                obj.coordCF = [t1;t2];
16                obj.ClearanceAngle = Theta;
17            end
18        end
19    %------------------Finding the rake face----------------------------
20        [H, THETA, RHO] = hough(obj.BW,'Theta',81:85);%Hough transformation
21        P  = houghpeaks(H, 10);
22        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
23        l = length(obj.lines);
24        obj.coordRF = [];
25        for i=1:l
26            Theta=obj.lines(i).theta;
27            t1 = obj.lines(i).point1;
28            t2 = obj.lines(i).point2;
29            rho = obj.lines(i).rho;
30            if rho < 103 && rho > 98
31                obj.coordRF = [t1;t2];
32                obj.RakeAngle = 90 - Theta;
33            end
34        end
35    end
```

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

The fixed position of tool allows also the predetermination of the ρ parameter, which is distance of the detected lines from the reference. 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 endings coordinates of the detected line and also the angle of the corresponding angular coefficient.

4.1.3.3 Rake and clearance face

With the data provided by the output of hough function, it is possible to extend the lines to match the entire rake and clearance edge. This is an important step of the analysis method because it allows to build an object (binary image) that is a mask to remove only the region of interest, the tool shape in this case . Consequently, it will be possible analyze the temperature fields and thermal behavior inside the tool without any interference from the temperatures in the vicinity.

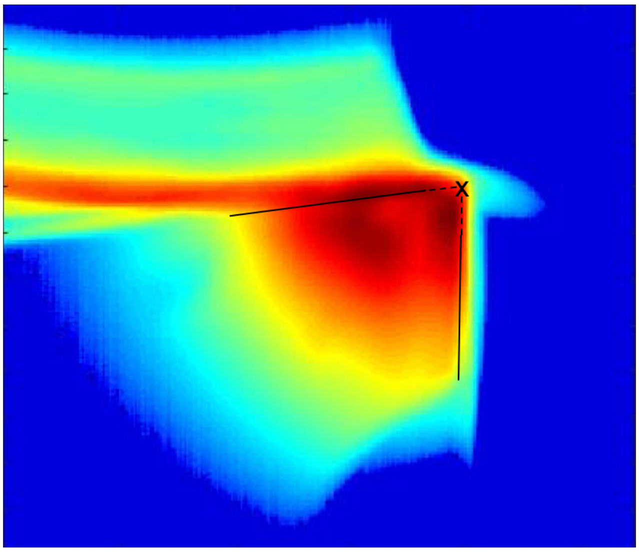

FIGURE 4.4 – Lines detected by hough transformation method

4.1.3.4 Tool tip coordinates

As the rake and clearance edges are determined, the tool tip will be calculated by means of the intersection between these lines. On the figure 4.4 the found lines are extended until they intersect, then the coordinates of tool tip 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

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

4.1.3.6 Temperature fields

In this step, it will be used the auxiliary function mentioned on the subsection Contour plot. This is an important function to determine same level curves, as the isotherms inside the tool shape. The contour levels are determined in a step of 50 ^{o}C.

```
            [C,h] = contour(obj.frame,v);
```

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, which are very valuable when comes to calculate heat flows.

```
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 it is representing and numxy is the amount of coordinates used to build the level. The coordinates are represented in the pair (x,y).

4.1.3.7 Heat flows - Chip and Tool

As described on section 3.2, the heat flow through tool and the energy carried away by chip are calculated. For heat flow through the tool, it is possible to extract isothermal lines by means of contour command and to calculate the gradient of temperatures with gradient command, which already is normal to the isothermal lines due to its properties. The width is already known 3.1. The length of the chosen isotherm is done by counting the amount of pixels provided by the coordinates in contour plot and turned into millimeter with the scale factor afterwards. In the case of the energy carried away by chip, the chosen line is placed on the end of contact chip - tool. The explanation for it 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 3.2, these values can be combined with the total power (P) generated during the cutting process to calculate the energy that goes to the workpiece by means of energy balance (equation 3.7). Then, it is possible to calculate the heat partition relative to each zone of interest.

$$p_i = \frac{\dot{Q}_i}{P} \tag{4.1}$$

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

4.2 Method validation

As described in the previous section Code implementation, there are many outputs of the implemented method as shear and normal stresses related to the mechanical part. However, in this paper the thermal modeling will be the focus of the discussions. The total power produced along this high speed machining was calculated as in the equation 3.3, the values are shown on figure 4.5.

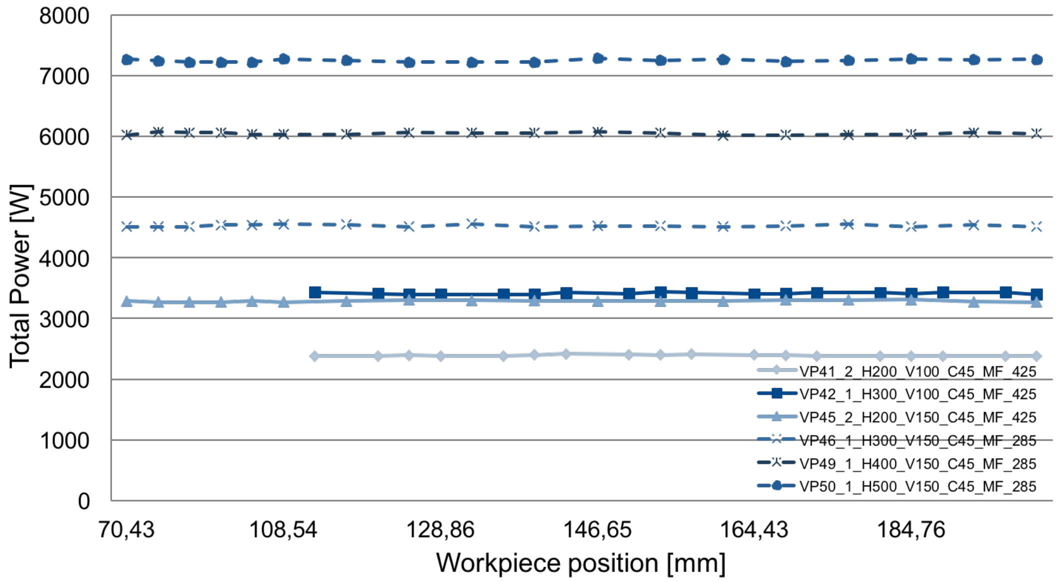

FIGURE 4.5 – Total power produced

As expected, the higher are the values of cutting velocity or depth of cut higher are the values of 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, it can be observed the thermal behavior of every area of interest along the workpiece position. 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 $100m/min$.

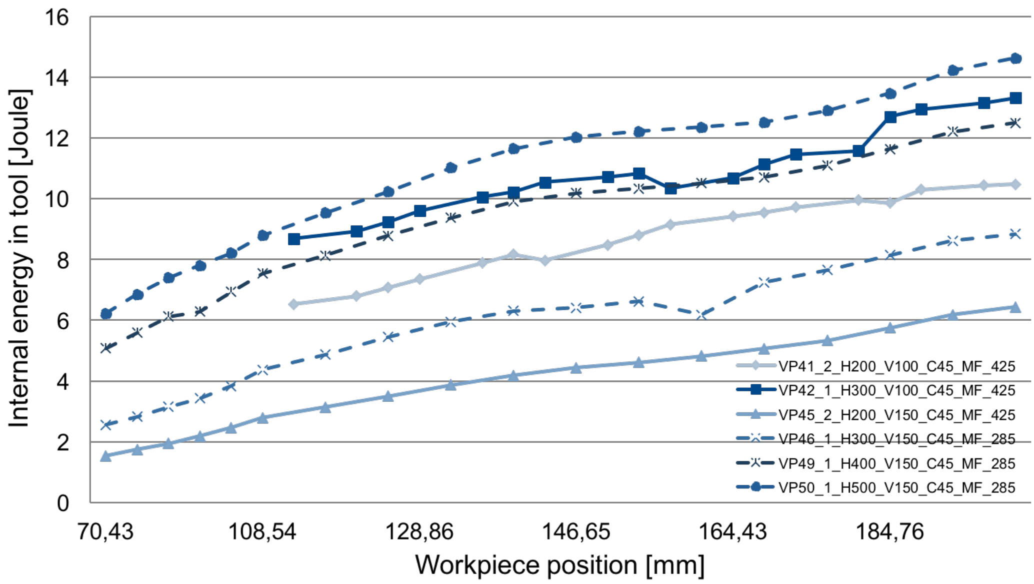

FIGURE 4.6 – Inner energy of tool along workpiece position

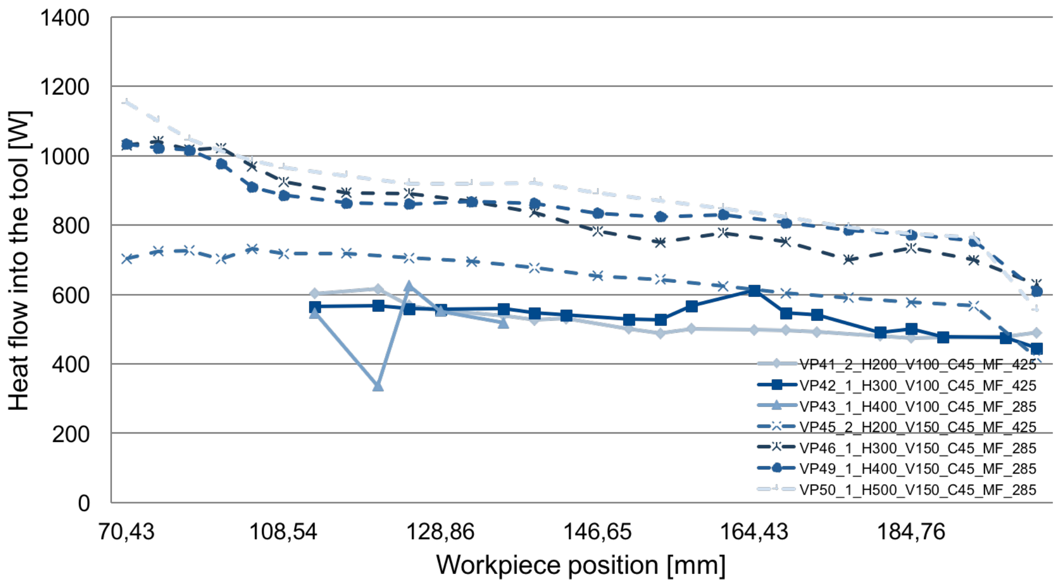

FIGURE 4.7 – Heat flow into tool

As it can be observed on the previous figure 4.7, the change rate of the inner energy of tool begins with a higher value than in the end of process. The rate starts to stabilize, indicating the beginning of the steady state.

To exemplify the results, it will be taken to represent the outcomes regarding heat partitions the experiment with cutting velocity $v_c = 150 m/min$ and depth of cut $a_p = 500 \mu m$. All the others experiments had approximately the same behavior during the cutting process. Concerning the heat partition through tool, workpiece and the energy carried away by chip, their behaviors can be observed on the figure 4.8. There is a slight decrement in the heat flow through tool, which it was expected due to the steady state as discussed before. As for the energy carried by chip, it may be noticed a slight increment.

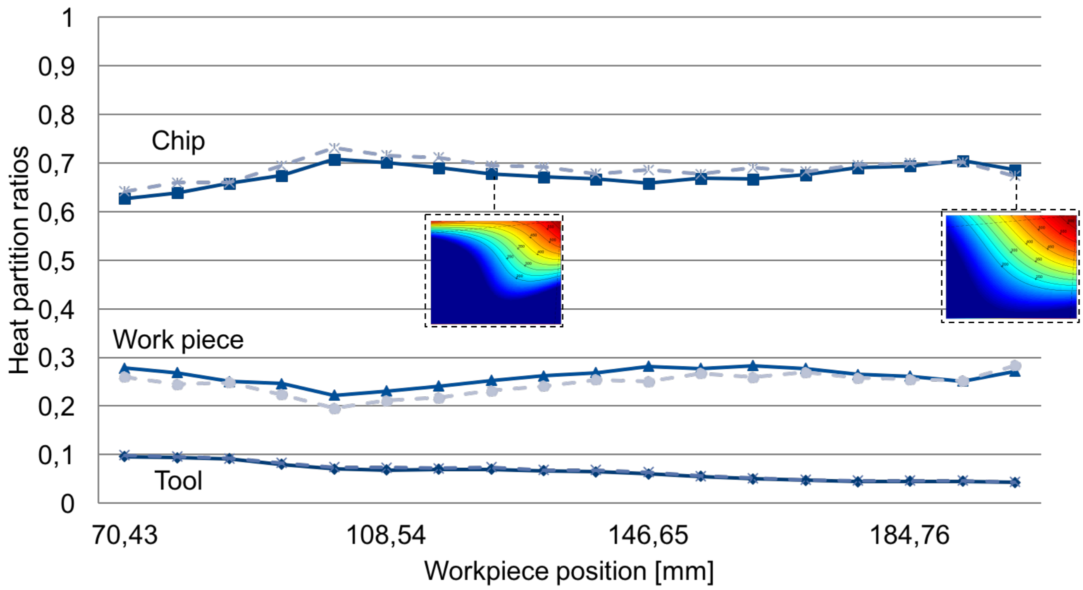

FIGURE 4.8 – Heat partition for experiment with $a_p = 500\mu m$ and $v_c = 150$ m/min

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. Also, it must be noticed the amount of energy that goes to chip (figure 4.9 and 4.10). The chip takes around 70% of the total energy produced, this fact may be explained due to the high temperatures that the region can reach and the high velocity of flowing.

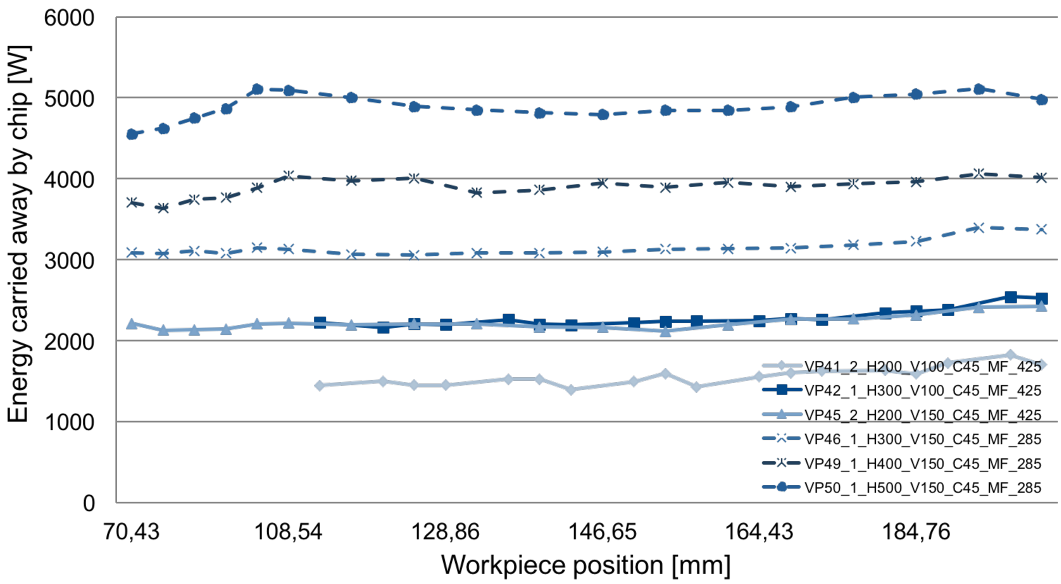

FIGURE 4.9 – Thermal energy into chip

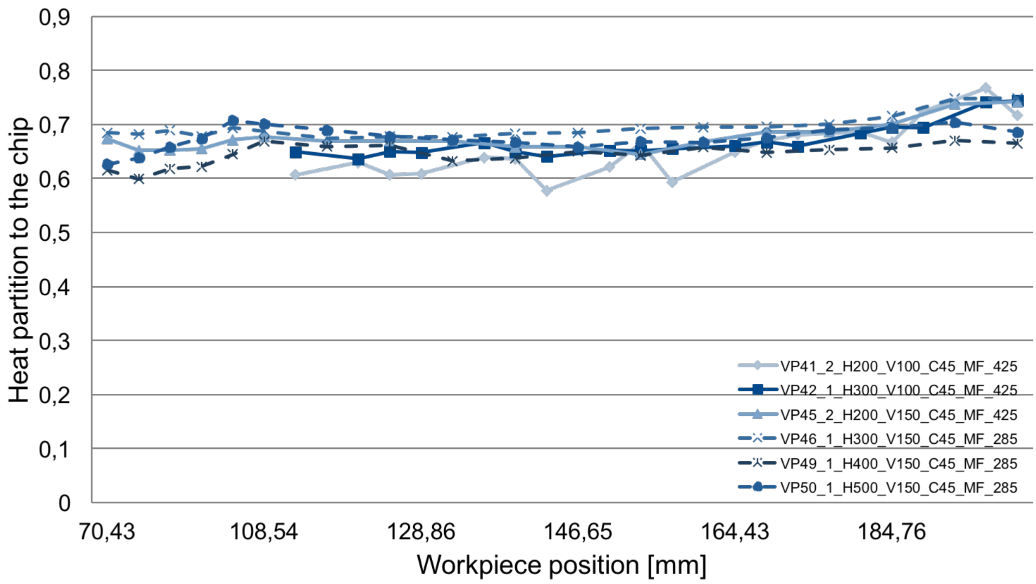

FIGURE 4.10 – Heat partition ratio for chip

As for the tool (figure 4.11), the partition of energy reaches a much smaller range when close to the steady state. The values of the partition to tool in this stage goes from 4% until 8%.

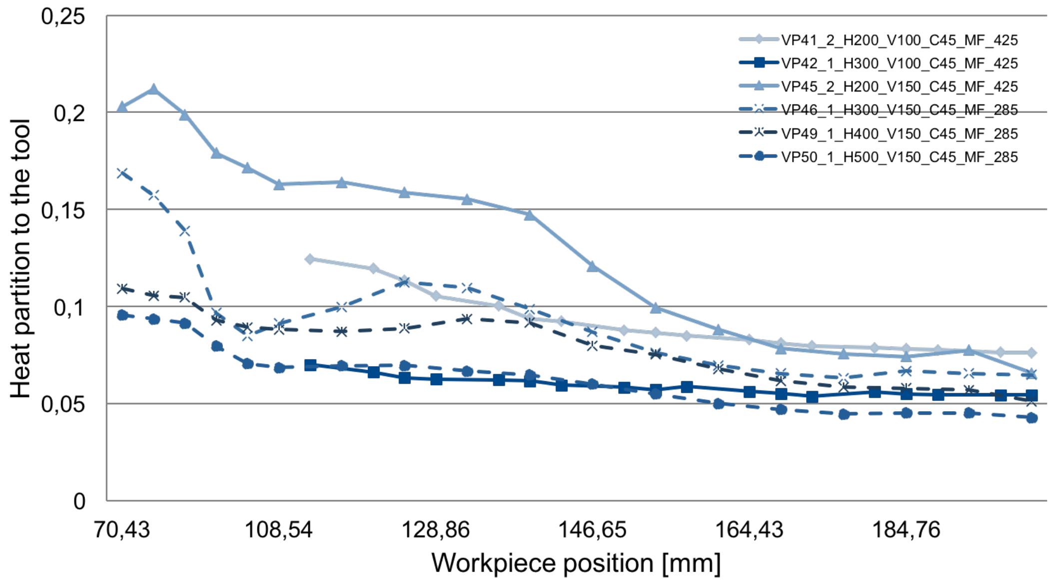

FIGURE 4.11 – Heat partition ratio for tool

5 Conclusions

The results found when processing thermal images provided a reasonable understanding about heat distribution through tool and chip components. Most of the heat generated during the cutting process goes to dissipation on the removed chip, about 70% of the total power generated. 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 suggests that this computational method also may be extended for this part of the process.

One of the problems to elaborate this work was that many of the videos were damaged due to pieces of chip interfering on the ideal presentation 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 impossible the use of some frames from the same video and entire other experiments sometimes.

It may be noticed that for experiments with the same relation $v_c \times a_p$ (figure 4.9) seems to provide the same energy to be carried by the flowing chip. Since the thermography method is very sensible to external interference and many experiments were damaged as mentioned before, it would be necessary to perform new experiments to validate this hypothesis.

The thermography method for temperatures measurement still presents some challenges, mainly when comes to set the correct emissivity. Even coating the tool and workpiece with black ink and carrying experiments to determine the emissivity of it, the ink cracks close to 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 filter of camera 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 image recognition patterns and image processing, is being used each time more nowadays processes. For a future study beyond this paper, computer vision can become an even stronger tool when combined with machine learning, which is revolutionizing most diverse areas. The principles used to build this computational method could be converted to analyze others types of cutting processes, as milling. Then, it could be turned into an intelligent system to support machining processes, improving all cutting parameters in order to obtain higher efficiency of tool, increasing tool life, better surface finishing of the workpiece and lower cutting time.

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.

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.

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

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.

TRIGGER, K. 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
    % This class was built to analyze the temperature inside the tool
    % shape, the temperature gradient, the isotherms...

    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;
        InternalEnergyTool;
        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;
        pointCF;%auxiliar to plot the cutting edge
        pointRF;
        pointM;
        Tx;%auxiliar to plot the gradients of the frame
        Ty;
        biImageTool;%Binary image of the tool shape
        biImageChip;
```

```
        biShearLine;
        xyMaxTemp;%coordinates of the point inside the chip with maximum Temperature
        lineChip;
        lineTool;
        validTemperature;
        heatCapacity;
        nExcPoints;
        heatAccumulatedPerLine;
        ptosLines;
        extPtosLineChip;
        line200;
    end

    methods
        % methods, including the constructor are defined in this block

        function obj = TemperatureAnalyze2(Frame,index)%constructor
            %Inputs -----------------------------------------
            Fp =  3220;%Cutting force in the power direction (Newtons)
            Fq = 1120;%Passive force (Newtons)
            widthTool = 4.4*10^-3;%meters
            Vp = 150/60;%meters/second
            tuc = 500*10^-6;%meters
            clength = 0.00251;%Define as an empty vector if we do not have
            %the mean value
            tt = [197 78];
            obj.validTemperature = 200;% For any experiment
            A = 0.1;%percentage of the deformation energy that is converted in heat
            %--------------------------------------------------
            obj.CuttingVelocity = Vp*60;%m/minute
            obj.UnCutChipThickness = tuc;
            obj.frame = Frame(index).f;
            if isequal(clength,[])
                clength = obj.contactLength();
            end
            obj.ContactLength = clength;
%             obj = obj.calculateCoordinates();
%             if isempty(obj.coordRF)==0 && isempty(obj.coordCF)==0
% %                 obj = obj.coordinateToolTip();
%             else %Default conditions
%                 if isempty(obj.coordRF)
%                     obj.RakeAngle = 6;
%                 end
%                 if isempty(obj.coordCF)
%                     obj.ClearanceAngle = 3;
%                 end
%             end
            obj.CoordinateToolTip = tt;
            obj.ClearanceAngle = 3;
            obj.RakeAngle = 6;
            obj.frame = Frame(index).f;
            obj = obj.toolContour();
            obj = obj.findLineTool();
            obj = obj.chipContour();
            obj = obj.findLineChip();
            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,widthTool);
            obj = obj.internalEnergyTool(widthTool);
            obj = obj.shearLine();
            obj = obj.calculatePecletNumber();
            obj = obj.forcesValues(Fp,Fq,widthTool,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)
            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)
            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)
            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;
        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];
        end

        function l = contactLength(obj)
            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? ');
            close all
            l = 15*10^-6*v;
        end

        function obj = maxTemperatureTool(obj)
            C = obj.biImageTool;
            Frame = C.*obj.frame;
            T = max(max(Frame));
            obj.MaximumTemperatureTool = T;
        end

        function obj = maxTemperatureChip(obj)
            Frame = obj.biImageChip.*obj.frame;
```

```
            obj.MaximumTemperatureChip = max(max(Frame));
        end

        function obj = meanTemperatureTool(obj)
            B = obj.biImageTool;
            Frame = B.*obj.frame;
            B = Frame > obj.validTemperature;
            Frame = B.*Frame;
            s = sum(sum(Frame));
            n = sum(sum(B));
            meanT = s/n;
            obj.MeanTemperatureTool = meanT;
        end

        function obj = displayBinary(obj)
            imshow(obj.BW);
            hold on
            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)
            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);
        end

        function obj = calculateCoordinates(obj)
            obj.BW = edge(obj.frame,'sobel');
        %-------------------Finding the clearance face----------------------
            [H, THETA, RHO] = hough(obj.BW,'Theta',2:5);%Hough transformation
            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
                Theta = obj.lines(i).theta;
                t1 = obj.lines(i).point1;
                t2 = obj.lines(i).point2;
                rho = obj.lines(i).rho;
                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
            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;
                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)
            figure
            imagesc(obj.frame);
            hold on
            plot(obj.CoordinateToolTip(1),obj.CoordinateToolTip(2),'xm')
            hold off
        end

        function obj = pointsRFandCF(obj)
            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;
            l1 = round(abs(obj.CoordinateToolTip(1)-extRF(1)));%length in pixels
    rake line
            l2 = round(abs(obj.CoordinateToolTip(2)-extCF(2)));%length in pixels
    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')
            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)'];
        end

        function obj = extremePointsChip(obj)
            [y,x] = find(obj.lineChip);
            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 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)
            [C,h] = contour(obj.frame,v);
            h.LineColor = [0.247 0.247 0.247];
            clabel(C,h,'manual','FontSize',10);
            x = obj.CoordinateToolTip(1);
            y = obj.CoordinateToolTip(2);
            axis([x-180 x+15 y-60 y+130])
            cb = colorbar('vert');
            zlab = get(cb,'ylabel');
            set(zlab,'String','Temperature (?C)');
            cb.Limits = [0 450];
            cb.FontSize = 10;
            zlab.FontSize = 10;
            daspect([1,1,1])
            ax = gca;
            v = [0.2 0.6 1.0 1.4 1.8 2.2 2.6];
            vt = v/0.015;
            vx = x + 15 - vt;
            ax.XTick = fliplr(vx);
            ax.XTickLabel = fliplr(v);
            ax.XAxisLocation = 'top';
            vy = y - 60 + vt;
            ax.YTick = vy;
            ax.YTickLabel = v;
            ax.YAxisLocation = 'right';
            xlabel('milimeters')
            ylabel('milimeters')
            hold off
        end

        function obj = calculateGradient(obj)
            pp = 15*10^-6;
            tx = zeros(size(obj.frame));
            ty = zeros(size(obj.frame));
            k = 0;
            for j = 1:5
                [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)
            auxx = [obj.pointCF(1) obj.CoordinateToolTip(1) obj.pointRF(1)];
            auxy = [obj.pointCF(2) obj.CoordinateToolTip(2) obj.pointRF(2)];
            k = 75.4;
            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 = 75.4;
            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)
            [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;
        end

        function obj = findLineTool(obj)
            [m,n] = size(obj.frame);
            Tmax = max(max(obj.biImageTool.*obj.frame));
            Tv = obj.validTemperature;
```

```
            v = round(Tv:40:Tmax);
            if length(v) == 1
                v = round([Tv Tmax]);
            end
            [C,~] = contour(obj.frame,v);
            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
                    obj.line200 = C(:,J+1:J+C(2,J));
                end
                B(:,:,k) = B(:,:,k).*obj.biImageTool;
            end
            obj.lineTool = B;
        end

        function obj = heatBalance(obj,tuc,Vc,w)
            k = 75.4;%heat conductivity
            pp = 15*10^-6; %pixel pitch
%---------------------------------------------------------------------------
            %First part - Heat carried away by the chip
             cp = [-4.39956806034758e-07 0.000707314520321484...
             -0.0488770693887544 481.214007868631]; %AISI 1045
            %Heat capacity for the workpiece
            M = obj.lineChip.*obj.frame;
            MH = polyval(cp,M);
            MH(MH == cp(4)) = 0;
            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
%             Vchip = 100*200/(60*n*15);
            p = 7874; %kg/m^3
            Qc = Hc*Vc*tuc*p;%Vc*tuc is the same for Vchip*tchip
            obj.HeatCarriedAwayByChip = Qc*w;
%---------------------------------------------------------------------------
            %Second part - Heat carried away by the tool
            dT = ((obj.Tx).^2 + (obj.Ty).^2).^(1/2);
            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*w*k;
            end
            obj.heatAccumulatedPerLine = Q;
            Qm = mean(Q(1:2));
            obj.HeatFluxAwayFromToolTip = Qm;
        end

        function n = exceedingPoints(obj,Temperature)
            B = obj.frame.*obj.biImageTool > Temperature;
            n = sum(sum(B));
        end

        function obj = internalEnergyTool(obj,w)
            pp = 15*10^-4;%in cm
            cp = [2.50542895559373e-10 -1.99579761670655e-06 0.00274369536032376
    3.09265830398264];%J/(K*cm3)
            %Heat capacity for tool
            Te = 22;
            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)*100;%Heat Amount for each pixel(J/m)
            Ha = sum(sum(H));%Mean value for the entire tool
            obj.InternalEnergyTool = Ha*w;
        end

        function B = passBinaryImageTool(obj)
            B = obj.biImageTool;
        end

        function B = passBinaryImageChip(obj)
            B = obj.biImageChip;
        end

        function obj = shearLine(obj)
            B = obj.biImageChip;
            v1 = sum(B);
            v1(v1 == 0) = [];
            l1 = length(v1);
            C = imcrop(B,[20 20 l1 100]);
            [m,n] = size(C);
            pto = zeros(1000,2);
            count = 1;
            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
            for i =1000:-1:1
                if isequal(pto(i,:),[0 0]) == 1
                    pto(i,:) = [];
                end
            end
            l = size(pto,1);
            for i = 1:l
                C(pto(i,1),pto(i,2)) = 0;
            end

            [H, THETA, RHO] = hough(C,'Theta',-40:-30);%Hough transformation
            P  = houghpeaks(H, 5);
            lin = houghlines(C, THETA, RHO, P, 'FillGap', 15,'MinLength',10);
            l=length(lin);
            p1 = [];
            p2 = [];
            for i=1:l
                Theta=lin(i).theta;
                t1 = lin(i).point1;
                t2 = lin(i).point2;
                y = abs(t1(2)-t2(2));
                if isempty(p1) && isempty(p2) && abs(Theta + 34) < 5
                    p1 = t1 + [19 19];
                    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)
                obj.ShearAngle = 30;
            end
        end

```

```
        function obj = forcesValues(obj,Fp,Fq,w,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 = w*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*w);%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.heatCapacity,obj.MaximumTemperatureCuttingZone);
            k = 75.4;
            d = 7.85*10^3;
            obj.PecletNumber = ((obj.CuttingVelocity/60)*obj.UnCutChipThickness)/(k/(cp*d));
        end

        function vH = displayHeatCumulateperLine(obj)%Fix this function
            d = zeros(size(obj.ptosLines,1),1);
            d2 = zeros(size(obj.ptosLines,1)-1,1);
            pp = 15*10^-3;%mm/pixel
            for i = 1:size(obj.ptosLines,1)
                d(i) = pp*((obj.CoordinateToolTip(1) - obj.ptosLines(i,1))^2 + ((obj.CoordinateToolTip(2) - obj.ptosLines(i,2))^2))^(1/2);
            end
            for i = 1:size(obj.ptosLines,1)-1
                d2(i) = pp*((obj.ptosLines(i+1,1) - obj.ptosLines(i,1))^2 + ((obj.ptosLines(i+1,2) - obj.ptosLines(i,2))^2))^(1/2);
            end
            d2 = mean(d2)*10^-3;
            figure
            plot(d,obj.heatAccumulatedPerLine,'-x')
            hold on
            q = gradient(obj.heatAccumulatedPerLine,d2);
            plot(d,q,'-*r')
            vH = [d obj.heatAccumulatedPerLine];
        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	54

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 each time more to guarantee the quality of processes. In the machining field there are many important parameters to assure that a process arranges the designed results. The superficial finishing of a workpiece and the tool life are some examples subjected to the direct influence of thermal energy generated in the heat zones. Due to it, there are a lot of theoretical methods for temperature modeling along the cutting zone, but still there is a lack of ways able to allow practical validation of these methods. Although many challenges still prevail on the suitable use of thermography, this technology makes possible 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 image processing toolbox support. It makes thermal image analysis, providing results about 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**