A STUDY OF THE MACHINABILITY OF ADDITIVELY MANUFACTURED INCONEL 625

A Thesis by

Rajesh Kumar Ananda Kumar

Bachelor of Engineering, Panimalar Engineering College, Anna University, 2014

Submitted to the Department of Industrial, Systems, and Manufacturing Engineering and the faculty of the Graduate School of Wichita State University in partial fulfillment of the requirements for the degree of Master of Science

December 2017
A STUDY OF THE MACHINABILITY OF ADDITIVELY MANUFACTURED INCONEL 625

The following faculty members have examined the final copy of this thesis for form and content, and recommend that it be accepted in partial fulfillment of the requirement for the degree of Master of Science, with a major in Industrial Engineering.

Wilfredo Moscoso, Committee Chair

Viswanathan Madhavan, Committee Co-Chair

Krishna K. Krishnan, Committee Member

Rajeev Nair, Committee Member
DEDICATION

To my parents, my brother, my friends, and my beloved wife
ACKNOWLEDGEMENTS

I would like to thank my advisor, well-wisher Prof. Wilfredo Moscoso-Kingsley for his continued support and guidance throughout my study at Wichita State University. I appreciate his patience while teaching or explaining me any concept. I would also like to thank my advisor, mentor Prof. Viswanathan Madhavan for sharing his immense knowledge in the subject and for his encouragement and feedbacks which helped me to carry out the research. I would like to extend my gratitude towards my committee members Profs. Krishna K. Krishnan and Rajeev Nair for their time and patience.

Special thanks to my friend, my colleague Arvind Natarajan, who has been a great support and encouragement from the day I joined the Advanced Manufacturing Processes Laboratory (AMPL). Our discussions on any topic helped me to think in another dimension which ultimately resulted in a better understanding. I am thankful to Homar, the man with lot of ideas and lot of knowledge. I appreciate his helping tendency and he is the main reason for keeping the lab active and fun-filled all the time. Thanks for introducing and teaching me to use MATLAB.

I would also like to thank my other colleagues Suraj, Cui, and Joshua for giving me the opportunity to collaborate and share knowledge. Thanks to my friends Navin, Kaushik, Arun, Bala, Nisha, Bharadwaj, Vetrivel, Vivek, Pavithra, Vivin, Phani, Lakshmi, Priyanka, Nikhil and Deepika for offering me advise and support. The dinners, outings, night games, group cookings and all the general help are appreciated.
ABSTRACT

Additively made parts will likely require post-processing by machining and other methods in order to improve surface finish and dimensional accuracy. This new class of materials have unique microstructures and properties that may impose new challenges during finishing operations like machining. Thus, a systematic study of the machining behavior of additively manufactured materials will provide data for the evaluation of their machinability. The objective of this thesis is to compare the machining behavior of additively manufactured Inconel 625 (AM IN 625) with that of a traditionally made, cast-wrought counterpart (CW IN 625). The comparisons will be made in terms of machining force, temperature at the chip-tool interface, and chip formation mechanisms and resulting morphology. A second objective is to evaluate the benefits of cryogenically cooling the tool and the work material using liquid nitrogen; specifically, evaluating the temperature at the chip-tool interface under such conditions.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. OVERVIEW</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.2 State of the Art</td>
<td>3</td>
</tr>
<tr>
<td>2. PROBLEM STATEMENT</td>
<td>7</td>
</tr>
<tr>
<td>3. MECHANICS OF MACHINING</td>
<td>8</td>
</tr>
<tr>
<td>4. WORK MATERIALS</td>
<td>10</td>
</tr>
<tr>
<td>5. METHODOLOGY</td>
<td>15</td>
</tr>
<tr>
<td>5.1 Experimental Configuration</td>
<td>15</td>
</tr>
<tr>
<td>5.2 Experimental Conditions</td>
<td>23</td>
</tr>
<tr>
<td>5.3 Data Processing</td>
<td>24</td>
</tr>
<tr>
<td>6. RESULTS</td>
<td>27</td>
</tr>
<tr>
<td>6.1 Machinability of AM IN 625 vs. CW IN 625</td>
<td>27</td>
</tr>
<tr>
<td>6.1.1 Temperature field</td>
<td>27</td>
</tr>
<tr>
<td>6.1.2 Machining forces</td>
<td>35</td>
</tr>
<tr>
<td>6.1.3 Tool Wear</td>
<td>41</td>
</tr>
<tr>
<td>6.1.4 Chip Morphology</td>
<td>48</td>
</tr>
<tr>
<td>6.2 Cryogenic Machining</td>
<td>49</td>
</tr>
<tr>
<td>7. DISCUSSION</td>
<td>55</td>
</tr>
<tr>
<td>8. CONCLUSIONS</td>
<td>61</td>
</tr>
<tr>
<td>9. FUTURE WORK</td>
<td>63</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>64</td>
</tr>
<tr>
<td>APPENDICES</td>
<td>68</td>
</tr>
</tbody>
</table>

A. Temperature data processing matlab code | 69 |
B. Force data processing matlab code | 88 |
C. Cast wrought IN 625 material certificate | 113 |
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Chemical composition of Inconel 625 powder used to fabricate the specimen</td>
<td>11</td>
</tr>
<tr>
<td>2.</td>
<td>Process parameters and machine setting used to fabricate the specimens</td>
<td>13</td>
</tr>
<tr>
<td>3.</td>
<td>Chemical composition of CW IN 625 rods</td>
<td>13</td>
</tr>
<tr>
<td>4.</td>
<td>Comparison of mechanical properties of AM IN 625 and CW IN 625</td>
<td>14</td>
</tr>
<tr>
<td>5.</td>
<td>Black-body calibration</td>
<td>20</td>
</tr>
<tr>
<td>6.</td>
<td>Summary statistics comparing CW IN 625 and AM IN 625</td>
<td>35</td>
</tr>
<tr>
<td>7.</td>
<td>Summary of tool wear in terms of volume loss per unit width of the tool</td>
<td>41</td>
</tr>
<tr>
<td>8.</td>
<td>Summary of tool wear in terms of volume loss per unit width of the tool in air atmosphere and LN atmosphere experiments</td>
<td>53</td>
</tr>
<tr>
<td>9.</td>
<td>Summary statistics comparing air atmosphere cuts with liquid nitrogen cuts</td>
<td>54</td>
</tr>
<tr>
<td>10.</td>
<td>Temperature vs emissivity for a range of radiation intensity</td>
<td>59</td>
</tr>
</tbody>
</table>
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Schematic of the 2-D orthogonal cutting.</td>
<td>9</td>
</tr>
<tr>
<td>2.</td>
<td>Schematic of Laser Powder Bed Fusion.</td>
<td>10</td>
</tr>
<tr>
<td>3.</td>
<td>Cylindrical specimen used to carry out the machinability study. The dimensions are in mm and the volume in cm³.</td>
<td>11</td>
</tr>
<tr>
<td>4.</td>
<td>The arrangement of the specimen in the build platform. Recoater direction is from left to right with respect to the figure. All the dimensions are in mm.</td>
<td>12</td>
</tr>
<tr>
<td>5.</td>
<td>Tensile specimen geometry as per ASTM E8/E8M standard.</td>
<td>14</td>
</tr>
<tr>
<td>6.</td>
<td>Schematic of the experimental setup for through-the-tool temperature measurement (Left). The actual setup (Right).</td>
<td>15</td>
</tr>
<tr>
<td>7.</td>
<td>Radiation intensity from black-body at temperatures from 700 °C to 982 °C. The solid orange curve is from the heating cycle and the dashed blue curve is from the cooling cycle.</td>
<td>21</td>
</tr>
<tr>
<td>8.</td>
<td>Experimental setup for cryogenic experiments.</td>
<td>23</td>
</tr>
<tr>
<td>9.</td>
<td>MATLAB GUI programmed to process the thermal data collected from the camera.</td>
<td>24</td>
</tr>
<tr>
<td>10.</td>
<td>MATLAB GUI showing the force processing.</td>
<td>26</td>
</tr>
<tr>
<td>11.</td>
<td>A) Magnified schematic of the cutting arrangement from the point of view of the camera. B) A sample image of radiation intensity as measured by the camera.</td>
<td>28</td>
</tr>
<tr>
<td>12.</td>
<td>Evolution of the temperature profiles after cutting CW IN 625. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2.</td>
<td>30</td>
</tr>
<tr>
<td>13.</td>
<td>Evolution of the temperature profiles after cutting AM IN 625. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2.</td>
<td>31</td>
</tr>
<tr>
<td>14.</td>
<td>Average temperature profile as a function of time for several machining velocities. Work material: CW IN 625. Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 μm/rev, cut width = 0.8 mm.</td>
<td>32</td>
</tr>
</tbody>
</table>
### LIST OF FIGURES (continued)

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>15. Average temperature profile as a function of time for several machining velocities. Work material: AM IN 625. Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, cut width = 0.8 mm.</td>
<td>33</td>
</tr>
<tr>
<td>16. Average temperature at the tool rake face as a function of distance from the cutting edge for a range of machining velocity between 1 m/s and 3 m/s. Work material: CW IN 625. Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, cut width = 0.8 mm.</td>
<td>34</td>
</tr>
<tr>
<td>17. Average temperature at the tool rake face as a function of distance from the cutting edge for a range of machining velocity between 1 m/s and 3 m/s. Work material: AM IN 625. Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, cut width = 0.8 mm.</td>
<td>34</td>
</tr>
<tr>
<td>18. Evolution of the peak temperature and the machining force after cutting CW IN 625. A) Machining velocity = 1 m/s; 2. B) Machining velocity = 2 m/s; C) Machining velocity = 3 m/s. The force labeled “X force” is the force perpendicular to the cutting and thrust forces. It is essentially “zero” as expected from orthogonal cutting.</td>
<td>37</td>
</tr>
<tr>
<td>19. Evolution of the peak temperature and the machining force after cutting AM IN 625. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2. The force labeled “X force” is the force perpendicular to the cutting and thrust forces. It is essentially “zero” as expected from orthogonal cutting.</td>
<td>38</td>
</tr>
<tr>
<td>20. Coefficient of friction as the function of time through the cut for a range of machining velocity between 1 m/s and 3 m/s. Work material: CW IN 625. Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, cut width = 0.8 mm.</td>
<td>39</td>
</tr>
<tr>
<td>21. Coefficient of friction as the function of time through the cut for a range of machining velocity between 1 m/s and 3 m/s. Work material: AM IN 625. Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, cut width = 0.8 mm.</td>
<td>40</td>
</tr>
<tr>
<td>22. Tool rake face after cutting CW IN 625. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2.</td>
<td>43</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES (continued)

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>23.</td>
<td>Tool rake face after cutting AM IN 625. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2.</td>
</tr>
<tr>
<td>24.</td>
<td>Surface profile of rake face after cutting CW IN 625. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2.</td>
</tr>
<tr>
<td>25.</td>
<td>Surface profile of rake face after cutting AM IN 625. A) Machining velocity = 1 m/s B) Machining velocity = 2 m/s C) Machining velocity = 3 m/s.</td>
</tr>
<tr>
<td>26.</td>
<td>Tool rake face wear profile after the machining. A), B), and C) corresponds to cuts with CW IN 625 at machining velocity = 1 m/s, 2m/s and 3 m/s resp. D), E), and F) corresponds to cuts with AM IN 625 at machining velocity of 1 m/s, 2m/s, and 3 m/s resp.</td>
</tr>
<tr>
<td>27.</td>
<td>a) Typical CW IN 625 chips, b) Typical AM IN 625 chips. V = machining velocity.</td>
</tr>
<tr>
<td>28.</td>
<td>Evolution of the temperature profiles. A) Machining in air atmosphere B) Machining in LN atmosphere without workpiece pre-cooling C) Machining in LN atmosphere with workpiece pre-cooling.</td>
</tr>
<tr>
<td>29.</td>
<td>Evolution of the peak temperature and the machining force. A) Machining in air atmosphere; B) Machining in LN atmosphere without workpiece pre-cooling; C) Machining in LN atmosphere with workpiece pre-cooling. The force labeled “X force” is the force perpendicular to the cutting and thrust forces. It is essentially “zero” as expected from orthogonal cutting.</td>
</tr>
<tr>
<td>30.</td>
<td>Wear patterns on the cutting edge and the tool rake face after the machining. A) Machining in air atmosphere. B) Machining in LN atmosphere with workpiece pre-cooling.</td>
</tr>
<tr>
<td>31.</td>
<td>Average temperature at the tool rake face as a function of distance from the cutting edge under air atmosphere and LN cooling condition. Work material: CW IN 625. Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, V = 2 m/s, cut width = 0.8 mm.</td>
</tr>
<tr>
<td>32.</td>
<td>Average temperature profile while machining CW IN 625 and AM IN 625 for machining velocities 1 m/s, 2 m/s, and 3 m/s.</td>
</tr>
<tr>
<td>33.</td>
<td>Temperature profile and tool wear patterns superimposed on rake face images A) Corresponds to CW IN 625 B) Corresponds to AM IN 625. V = Machining velocity.</td>
</tr>
</tbody>
</table>
CHAPTER 1
OVERVIEW

1.1 Introduction

Hybrid manufacturing methods combining additive manufacturing (AM) and subtractive manufacturing (SM) have emerged as potentially viable processes to obtain fully functional parts with acceptable mechanical properties, dimensional accuracy and surface integrity [1]. Although AM by itself offers access to complex geometries and efficient material utilization, it is handicapped by dimensional accuracy and surface integrity [1]. The poor dimensional accuracy and surface integrity resulting from AM processing demands finishing by subtractive routes such as machining. However, these AM materials have unique microstructures and properties that may impose new challenges during these finishing operations. Thus, a systematic study of the machinability of AM materials is required to design hybrid AM + SM methods.

The machinability of a material is affected by a number of inputs including machining force, chip formation mechanism, chip-tool contact friction, and subsurface thermo-mechanical damage; and ultimately tool temperature and wear [2]. Control of temperature and wear at the tool rake face, where they are usually most severe, is particularly elusive. To achieve control, temperature and wear must first be measured, and since temperature causes wear, the most fundamental measurement is that of temperature. Attempts to measure the temperature at the chip-tool interface have relied on methods such as (1) the tool-workpiece thermocouple technique [3, 4], (2) correlation of metallurgical changes in the tool material and temperature [5, 6], (3) implantation of micro-thermocouple arrays on the tool rake face [7], and (4) through-the-tool thermography [8]. Out of these methods, through-the-tool thermography promises low uncertainty, and high spatial and temporal resolution. The technique’s uncertainty decreases with increase in
the temperature of the target (the chip-tool interface), and is quite low (±10 °C) at ~1,000 °C [9], which is the expected temperature of the chip-tool interface when machining super-alloys. Therefore, it should be suitable to use thermography to measure the temperature at the chip-tool interface during the machining of super-alloys.

This thesis provides an evaluation of the machinability of Inconel 625 prepared by an AM route (AM IN 625). The material was prepared by laser powder bed fusion (LPBF), and was then utilized for the machining tests, which were performed by cutting along one particular direction in the material. The machining force (cutting and thrust forces), and the temperature of the chip-tool interface were observed in situ. Chip morphology and tool wear were observed post mortem. These parameters were compared to those obtained from the machining of a conventionally made, cast-wrought IN 625 (CW IN 625). The machining was performed with a specially designed tool. This tool, a transparent yttrium aluminum garnet (YAG) cube with 12 useable cutting edges, provided an optical path to the interface for the observation of the radiation emitted by the side of the chip in contact with the tool rake face (i.e., the chip underside). The radiation was converted into real chip-tool interface temperature after a calibration procedure. The comparisons show clear differences in chip formation mechanisms from AM and CW IN 625, and in the chip-tool interface temperature. The data may be used to inform the design of hybrid (LPBF + machining) processing. It also points in the direction in which future research should be conducted to optimize hybrid AM + SM processes, and in particular the hybrid LPBF + machining. The addition of liquid nitrogen (LN) as a cutting fluid has been found to reduce temperature, even when it has led to increased machining forces.
1.2 State of the Art

There are some instances that indicate that AM-made materials can be produced, after proper heat treatments, with mechanical properties often superior to those of the conventionally made, cast and wrought counterparts [10]. Indeed, due to rapid cooling of the small volumes locally heated, the microstructure of the AM material may be ultra-refined and martensitic, and therefore its strength may be larger than that of the conventionally-made material [11]. Many other reports, however, indicate the contrary. The consensus is that AM materials have mechanical properties that are distinct from those of conventional materials.

Jacob et al. [12] observed the mechanical strength and ductility of AM 17-4 PH steel as a function of source powder recycling. The results were the same when recycling the source powder up to ten times. The as-built AM steel had higher yield strength and lower ductility (1344 MPa and 26%, respectively) than those from ASTM-compliant cast-wrought counterparts (1034 MPa and 40%, respectively). The report attributes the results to the presence of a fully martensitic microstructure in the AM material. Whereas Rafi et al. [13] showed that AM-made 17-4 PH steel resulted in low strength and high ductility compared to a solution treated cast-wrought control sample, their microstructural analysis revealed that they had not been successful at producing a fully martensitic phase directly from the melt re-solidification. Mower et al. [11] studied the static and dynamic mechanical behavior of several materials processed by AM routes. For most materials, the tensile strength obtained directly from the printing was either lower or comparable to those from cast-wrought counterparts. However, the 316L steel that was synthesized had tensile strength of ~ 699 MPa, and the cast-wrought equivalent had a tensile strength of ~ 563 MPa. The higher strength was attributed to a finer microstructure. The elongation at failure was the same for both materials (~ 28% to 30%). Fatigue strength was generally lower in the AM-made material
than in the conventional material. Although hot isostatic pressing (HIP) increased fatigue strength, the fatigue performance of the AM materials was generally inferior to that of the conventional materials. The smaller fatigue strength was attributed to stress concentration at surface defects and internal microcracks and voids. Hollander et al. [14] reported the mechanical properties of AM Ti-6Al-4V and conventional Ti-6Al-4V. Their report showed tensile strength and elongation at breakage of ~ 1211 MPa and ~ 6.5%, respectively, for the as-built AM material, and ~1042 MPa and ~13%, respectively, for the cast-wrought counterpart. Explanations for the observed strength and ductility differences were not explicitly given.

The unique mechanical properties of AM materials may produce a negative impact on their machinability, and thus may challenge SM processing to obtain dimensional accuracy and surface integrity. An extra layer of complications faced by SM post-processing may arise from the significant strength anisotropy also observed in materials processed by AM routes [15], which is associated with directional solidification and columnar grain growth.

The complications that may arise during the machining of additively printed parts may be particularly adverse for parts made from super-alloys of titanium or nickel. As a result of the high strength and low thermal conductivity of these alloys, machining materials such as Ti6Al4V and IN 718 results in high tool temperatures and wear rates [7]. In addition, the low thermal conductivity of these materials results in adiabatic heating during their machining, which not only feeds back into the tool temperature and wear rate, but plays a role in the type of chip formation (continuous vs. serrated), and in the machining dynamics [16].

There are reports in the literature that indicate that there are great benefits associated with the use of non-conventional cutting fluids such as liquid nitrogen (LN) and cryogenic carbon dioxide (CO2) gas. It is generally seen that the application of coolant in a metal cutting process can
decrease the temperature and wear of the tool, and surface roughness of the workpiece; thus, increasing the productivity.

Cutting fluids act as lubricants by reducing the friction between the tool and the chip. The most common cutting fluids are water soluble and oil based. Water soluble fluids are mainly used during operations involving high cutting speeds and provide a cooling effect. In cases where a lubricating effect is desired, oil based fluids are used. Other than being harmful to the environment [17], it is also seen that cutting fluids are not effective in penetrating the chip-tool interface.

A material is classified as cryogenic if its temperature is below -150°C. Liquid nitrogen (LN) is most commonly used in many industries for cooling purposes. Liquid nitrogen has been used to precool the workpiece in a bath, flooding the cryogenic fluid on the tool and workpiece, or using micro nozzles to apply cooling effect on the chip tool interface. It was seen that tool wear decreases because of a decrease in temperature. But temperature drop of the workpiece can increase the shear stress which can lead to increased cutting force, thus increasing the tool wear and decreasing the tool life. [17]

The difference in temperature and cutting force while milling Inconel 718 using conventional oil based coolant and LN were studied by Ampara et al. [18]. The cutting temperature was measured using a thermocouple method for dry, conventional oil-based coolant, and LN. This work showed that the LN could penetrate the cutting zone. Thus, reducing the temperature. The force measurement showed that the LN had higher cutting and thrust force when compared to the conventional coolant, and lower when compared against dry machining. Kumar et al. [19] investigated the effect of LN on tool wear and cutting force while machining SS202 steel. The results showed that application of LN lowered the cutting force by 14.83% and lowered the flank
wear of the tool by 37.39%. The difference in the flank wear significantly increased at higher machining velocities.

In the work carried out by Jerold et al. [20], the effect of conventional cutting fluid, cryogenic CO\textsubscript{2}, and LN was studied and the cutting force, cutting temperature, tool wear, surface roughness, and chip morphology were reported. The application of CO\textsubscript{2} and LN resulted in temperature drop of about 6-21\% and 9-34\% respectively. The machining forces were reduced to a greater extent in case of CO\textsubscript{2} but not appreciable amount in the case of LN. Bermingham, M.J. [21] concluded that the thrust forces were always higher when cryogenic coolants were utilized. Also, there was no significant change in the feed force with cryogenic coolant. It is reported that the coefficient of friction did not reduce with cryogenic coolant. But in some cases, the coefficient of friction increased when cryogenic coolant was used.
CHAPTER 2

PROBLEM STATEMENT

The objective of this thesis is to compare the machining behavior of additively manufactured Inconel 625 (AM IN 625) with that of a traditionally made, cast-wrought counterpart (CW IN 625). The comparisons will be made in terms of machining force, temperature at the chip-tool interface, and chip formation mechanisms and resulting morphology. A second objective is to evaluate the benefits of cryogenically cooling the tool and the work material using liquid nitrogen; specifically, evaluating the temperature at the chip-tool interface under such conditions.

From the state of the art, it can be concluded that additively made parts will likely require post-processing by machining and other methods in order to improve surface finish and dimensional accuracy. It is also clear that additively made parts may be made with quasi-static strength that exceeds that of parts made from conventional cast-wrought materials; but with much smaller ductility. Thus, a systematic study of the machining behavior of the additively manufactured materials will provide data for the evaluation of strength and ductility, and their effect in machinability. The data may be utilized to tailor machining optimization methods for the unique case presented by the additively made materials so that hybrid additive-subtractive methods can be properly deployed.

The evaluation of machinability proposed herein, which will be based on a multi-dimensional parametric study, will give information to validate computational models that attempt to capture the complexity of the deformation zone during the chip formation process. These models usually attempt to match the variables that will be measured. That is, force, temperature distribution, chip thickness and chip type. The highly-instrumented setup to be adopted will provide the required measurements with high temporal and spatial resolution.
CHAPTER 3
MECHANICS OF MACHINING

Measuring the cutting force ($F_n$) and thrust force ($F_t$) (Figure 1) is essential for analyzing the machinability of any material. Merchant’s model (Figure 1) shows the relationship between kinematic variables and forces involved in 2-D orthogonal cutting [22]. The machining velocity is $V$ and the chip flow velocity is $V_c$. The feed per revolution of workpiece is $f$ and the chip thickness is $f_c$. The ratio of $V_c$ over $V$ is the cutting ratio, which is also equal to $f$ over $f_c$. The net force exerted by the tool in the machining process is $F_0$. The force $F_0$ can be resolved in two mutually perpendicular components in many ways, out of which two possible ways are shown in Figure 1. The resolved components of the force can be used to analyze the mechanics of the metal cutting operation. These forces are a function of the shear plane angle $\phi$, and rake angle of the tool $\alpha$. The various forces and the friction coefficient at the rake face are given by:

$$F_s = \text{shear force on shear plane} = F_n \cos \phi - F_t \sin \phi.$$  \hfill (3.1)

$$F_{ns} = \text{normal force on shear plane} = F_n \sin \phi + F_t \cos \phi.$$ \hfill (3.2)

$$F_f = \text{friction force on rake face} = F_n \sin \alpha + F_t \cos \alpha.$$ \hfill (3.3)

$$F_{nf} = \text{normal force on rake face} = F_n \cos \alpha - F_t \sin \alpha.$$ \hfill (3.4)

$$\mu = \text{friction coefficient at rake face} = \frac{F_f}{F_{nf}}.$$ \hfill (3.5)

The rake angle $\alpha$ of the tool is found from the design of the tool holder. The shear angle $\phi$ can be found by measuring the thickness of the chip formed during the machining process. The density of the work material is not changed during machining. Hence,

$$t_i w_i l_i = t_f w_f l_f,$$ \hfill (3.6)
where, $t_i, w_i$ and $l_i$ are the undeformed chip thickness or feed, width of the cut and length of the cut, respectively, and $t_f, w_f$ and $l_f$ are the chip thickness, width of the chip and length of the chip, respectively. For plane strain, the width of the cut and the chip are the same, so $w_i = w_f$. Hence, $w_i$ and $w_f$ in equation (3.6) drop out to give:

$$r = \frac{t_i}{t_f} = \frac{l_f}{l_i}$$  \hspace{1cm} (3.7)

where, $r$ is the cutting ratio. The shear angle is given by the equation,

$$\tan \varphi = \frac{r \cos \alpha}{1 - rsin \alpha}.$$  \hspace{1cm} (3.8)

The energy consumed in the process of metal cutting can be evaluated using all the parameters from the above set of equations. The total power consumed during a metal cutting process is given by

$$P = F_e V.$$  \hspace{1cm} (3.9)

Figure 1. Schematic of the 2-D orthogonal cutting.
CHAPTER 4

WORK MATERIALS

The additively manufactured Inconel 625 (AM IN 625) work specimens used in the study were manufactured at the National Institute of Standards and Technology (NIST). The details of the work specimen fabrication and mechanical properties are given in [27]. A generic illustration of the process, which was based on Laser Powder Bed Fusion (LBPF), is shown in Figure 2. The powder used for the LPBF was Inconel 625 (nickel super alloy). The powder conformed to the standards ASM 5666 and ASTM F3056-14e1, which are the standards for additively manufactured nickel super alloy Inconel 625 (UNS N06625). The chemical composition of the powder is tabulated in Table 1. The particle size of the powder used was less than 50 µm [27].

Figure 2. Schematic of Laser Powder Bed Fusion
TABLE 1

CHEMICAL COMPOSITION OF INCONEL 625 POWDER USED TO FABRICATE THE SPECIMEN [27]

<table>
<thead>
<tr>
<th>Element Content</th>
<th>Chemical composition (mass percentage)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carbon, C</td>
<td>≤ 0.1</td>
</tr>
<tr>
<td>Manganese, Mn</td>
<td>≤ 0.5</td>
</tr>
<tr>
<td>Phosphorus, P</td>
<td>≤ 0.015</td>
</tr>
<tr>
<td>Sulfur, S</td>
<td>≤ 0.015</td>
</tr>
<tr>
<td>Silicon, Si</td>
<td>≤ 0.5</td>
</tr>
<tr>
<td>Nickel, Ni</td>
<td>≥ 58</td>
</tr>
<tr>
<td>Chromium, Cr</td>
<td>20 to 23</td>
</tr>
<tr>
<td>Molybdenum, Mo</td>
<td>8 to 10</td>
</tr>
<tr>
<td>Cobalt, Co</td>
<td>≤ 1</td>
</tr>
<tr>
<td>Iron, Fe</td>
<td>≤ 5</td>
</tr>
<tr>
<td>Niobium Nb</td>
<td>3.15 to 4.15</td>
</tr>
<tr>
<td>Titanium</td>
<td>≤ 0.4</td>
</tr>
<tr>
<td>Aluminum</td>
<td>≤ 0.4</td>
</tr>
<tr>
<td>Tantanium</td>
<td>≤ 0.05</td>
</tr>
</tbody>
</table>

The cylinders made by AM using the Inconel powder are shown in Figure 3 and Figure 4. Table 2 gives the information about the process parameters and machine settings used to fabricate the AM cylinders.

![Section view A-A](image)

Figure 3. Cylindrical specimen used to carry out the machinability study. The dimensions are in mm and the volume in cm³.
Figure 4. The arrangement of the specimen in the build platform. Recoater direction is from left to right with respect to the figure. All the dimensions are in mm.

The cast-wrought Inconel 625 (CW IN 625) material used in the study was high-strength 625 nickel rod of 25.4 mm in diameter and 152.4 mm in length from VDM metals USA, LLC. The composition of the metal alloy, as provided by the manufacturer, is shown in Table 3. The rods were turned down to 20.28 mm OD, and ID boring was done to achieve a wall thickness of 0.9 mm. After machining the rods to the dimension matching that of AM IN 625, both workpiece specimens were annealed with the same procedure carried out by the manufacturer of the CW IN 625 rods.

In order to characterize the material properties of the additively manufactured specimens, tensile tests were carried out. The test results gave the elastic modulus (E), ultimate tensile strength (UTS), 0.2% offset yield strength (YS) and upper yield strength (UYS). Table 4 shows a summary of these parameters, and a comparison with the equivalent values for the cast-wrought counterpart.
The tensile tests were carried out as per the ASTM E8/E8M standard. The geometry of the tensile test specimens, made by the same LPBF used to make the cylindrical specimens of this study, are shown in Figure 5. The build direction during the LPBF of the tensile specimens was transverse to the tensile axis.

**TABLE 2**

PROCESS PARAMETERS AND MACHINE SETTING USED TO FABRICATE THE SPECIMENS [27]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scan Pattern</td>
<td>Striped</td>
</tr>
<tr>
<td>Stripe Width</td>
<td>4 mm</td>
</tr>
<tr>
<td>Laser Power (P_L) [W]</td>
<td>195</td>
</tr>
<tr>
<td>Scan Speed (v_L) [1000 mm · s⁻¹]</td>
<td>800</td>
</tr>
<tr>
<td>Layer Thickness (t_L) [mm]</td>
<td>0.02</td>
</tr>
<tr>
<td>Raster Line Separation / Hatch (h_L) [mm]</td>
<td>0.1</td>
</tr>
<tr>
<td>Atmosphere</td>
<td>N₂</td>
</tr>
</tbody>
</table>

**TABLE 3**

CHEMICAL COMPOSITION OF CW IN 625 RODS

<table>
<thead>
<tr>
<th>Element Content</th>
<th>Chemical composition (mass percentage)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carbon; C</td>
<td>0.034</td>
</tr>
<tr>
<td>Manganese; Mn</td>
<td>0.19</td>
</tr>
<tr>
<td>Phosphorus; P</td>
<td>0.005</td>
</tr>
<tr>
<td>Sulfur; S</td>
<td>0.001</td>
</tr>
<tr>
<td>Silicon; Si</td>
<td>0.214</td>
</tr>
<tr>
<td>Nickel; Ni</td>
<td>60.64</td>
</tr>
<tr>
<td>Chromium, Cr</td>
<td>21.87</td>
</tr>
<tr>
<td>Molybdenum; Mo</td>
<td>8.85</td>
</tr>
<tr>
<td>Cobalt, Co</td>
<td>0.05</td>
</tr>
<tr>
<td>Iron, Fe</td>
<td>4.17</td>
</tr>
<tr>
<td>Niobium Nb</td>
<td>3.48</td>
</tr>
<tr>
<td>Titanium</td>
<td>0.2</td>
</tr>
<tr>
<td>Aluminum</td>
<td>0.23</td>
</tr>
<tr>
<td>Tantanium</td>
<td>0.008</td>
</tr>
</tbody>
</table>
Figure 5. Tensile specimen geometry as per ASTM E8/E8M standard [27]

TABLE 4

COMPARISON OF MECHANICAL PROPERTIES OF AM IN 625 AND CW IN 625

<table>
<thead>
<tr>
<th>Property</th>
<th>AM IN 625 *</th>
<th>CW IN 625</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elastic modulus (GPa)</td>
<td>179</td>
<td>205</td>
</tr>
<tr>
<td>Yield stress (MPa)</td>
<td>0.78</td>
<td>0.49</td>
</tr>
<tr>
<td>UTS (MPa)</td>
<td>1.07</td>
<td>0.93</td>
</tr>
<tr>
<td>% elongation</td>
<td>36</td>
<td>52</td>
</tr>
<tr>
<td>Vickers hardness (kg/mm²)</td>
<td>330</td>
<td>320</td>
</tr>
</tbody>
</table>

* [27]
CHAPTER 5
METHODOLOGY

5.1 Experimental Configuration

The temperature distribution at the chip-tool interface was obtained after machining on a high rigidity and high speed, custom built lathe. The setup was designed to rotate the workpiece and feed it towards the stationary tool as in axial turning as shown in Figure 6. The lathe’s loop stiffness was in the order of 100 N/µm. The spindle speed capacity was 10,000 rpm. The cutting tool was made of an optically transparent yttrium aluminum garnet (YAG) crystal. The transparent tool and the mirror mounted under it enabled direct observation of the chip-tool interface, which was performed with an intensified, charged couple device; henceforth referred to as the ICCD camera. The machining was performed on the CW IN 625 and AM IN 625.

Figure 6. Schematic of the experimental setup for through-the-tool temperature measurement (Left). The actual setup (Right).

The YAG tools, supplied by Red Optronics, were 6-mm cubes having 12 usable cutting edges. To increase the toughness of the YAG cubes, they were annealed in a Sensor Tech Corp. STT-1600-2.75-12 high temperature tube furnace. The YAG cubes were heated at a rate of 4 °C
per minute, held at 1600 °C for 10 hours, and then cooled back to ambient temperature at a rate of
4 °C per minute. The edge radius, as received from Red Optronics, was about 2.5 μm. The YAG
cubes were held by a specially designed tool holder that allowed cutting at -5° rake angle and +5°
relief angle. The fixture supported the cubes on three mutually orthogonal faces. A clamp secured
the cubes in place. The tool holder had a small pocket under the cube support, where a 1.1-mm
thick glass front-surface mirror was mounted. As mentioned above, the mirror directed the
radiation emitted by the surface of the chip in contact with the tool rake face towards the camera.
An “off-the-shelf” tungsten carbide (TC) tool with rake angle = 0° and relief angle = 11° was also
used for some tests.

The workpieces were prepared by machining from solid cylindrical rods. The workpieces
were tubes having either 20.32 mm outer diameter (OD) and 0.9 mm thickness, or 25.4 mm OD
and 1 mm thickness. The tubes were held by a collet on the lathe that allowed a firm installation
with minimal run out. The axis of the workpieces was mounted parallel to the lathe spindle axis.
For the AM workpiece, this axis was also the direction of the build during the LPBF process (Figure
4). For the CW workpiece, the direction of the axis relative to the rolling direction is unknown,
but due to expected isotropy, the orientation is unlikely to affect the results. The experimental
configuration approximated orthogonal cutting at a machining velocity that was taken as π x
workpiece OD x spindle rpm. The solid cylinders had been annealed by the material’s supplier in
air at 1038°C for 5 minutes, followed by air cooling outside the furnace. For the 0.9-mm thick
workpiece, this annealing treatment was repeated after turning into tubes. These workpieces were
utilized for the machining experiments on both CW IN 625 and AM IN 625 that were carried out
in an air atmosphere. However, Vickers indentation showed that the second annealing did not have
an effect in hardness. The 1.0-mm thick workpieces were used for the cryogenic machining, and were annealed only while in solid-cylinder form by the material’s supplier.

The camera, a Princeton Instruments PI-MAX2 was coupled to a custom-built lens that incorporated a series of apertures/Lyot stops designed to mitigate size-of-source effects. The imaging system had an effective magnification of 5x, and a focal plane array (FPA) of 1024 x 1024 pixels. The pixels were squares having 12.8-μm long sides. Therefore, each pixel was equivalent to approximately a square region of the chip-tool interface having 2.56-μm sides. The total field of view was a square of area = 2621 μm x 2621 μm. The camera control software allowed grouping several pixels into one box – a technique known as “binning”. For binning, 16 pixels were combined in a 4 x 4-pixel square sub-array. In other words, the 1024 x 1024-pixel array was converted into a 1024/4 x 1024/4 = 256 x 256-pixel array. Each bin of pixels (for simplicity, henceforth referred to as a pixel) was then equivalent to a 10.24 μm x 10.24 μm area from the chip-tool interface. Therefore, the theoretical spatial resolution of the imaging system was 10.24 μm.

The experiments produced chip-tool contact lengths, measured from the cutting edge, along the rake face, of 100 to 150 μm. Therefore, along this length, the field included about 10 to 15 observation points. Binning was important to increase the temporal resolution of the imaging system, which was, after the binning, about 0.04 ms (about 25-Hz frame rate). The camera had a 16-bit dynamic range, which means that the maximum radiation intensity it could detect, in counts, was 65,535. The laboratory was darkened throughout the experiments to reduce background light noise to a minimum. The camera was used to record the background light noise prior to each cutting experiment. These images usually resulted in radiation intensities of about 200 counts. The mean background image was obtained from several frames, after computing average counts, pixel
by pixel. The mean background image was subtracted from all frames taken during machining, pixel by pixel.

In order to relate measured radiation intensity to chip-tool interface temperature two pieces of information are needed: 1) The relation between the radiation intensity read by the ICCD camera from the emission of a black-body and the temperature of the black-body (i.e., the calibration), and 2) the emissivity of the side of the chip in contact with the tool rake face (i.e., the chip underside). The former was performed by recording images of the cavity of an Omega BB-4A black-body, after heating and cooling it through temperatures ranging from 700 to 982 °C. As with the actual machining experiments, the camera was used to record the background light noise prior to each calibration experiment. The mean background image was subtracted from all frames taken during the calibration, pixel by pixel. The latter was estimated from information compiled from [26], which reported the emissivity of IN 625 under conditions approximating the chip underside. That is, for non-oxidized, smooth surfaces.

The radiation intensity from the black-body, after background light noise removal, under a range of camera exposure time values, is shown in Table 5. The data suggests that, for a source at a given temperature, the radiation intensity per unit exposure time was a constant. This fact enabled the use of a single calibration curve to relate radiation intensity and temperature for all exposure time values. Note also that the camera responds with minimal hysteresis (< 0.5%), so there is no bias in reading increasing or decreasing temperatures. The calibration was obtained by fitting the black-body radiation intensity to temperature data to the forward Sakuma-Hattori equation. The equation has the following form:

\[
F(T) = \frac{C}{\exp\left(\frac{c_2}{AT + B}\right) - 1}
\]  

(5.1)
where \( F(T) \) is the radiation intensity from the black-body (counts), \( T \) is the black-body temperature (K), \( c_2 \) is the second radiation constant (14,388 \(\mu\)m-K), and \( C, A \) and \( B \) are constants. The fit returns the values of the constants, which for the imaging system described herein are \( C = 5.15E+14 \) counts, and \( A = 0.4 \) \(\mu\)m and \( B = 135 \) \(\mu\)m-K. The maximum standard deviation from black-body radiation intensity measurements is 713 counts, and the Sakuma-Hattori equation fits the data with a goodness of fit greater than 99.9%.

To estimate the true temperature of the chip underside, two corrections have to be introduced. The first correction is the emissivity of the chip underside, which is generally less than 1. From [26] the emissivity of IN 625 under conditions approaching those of the chip underside is about 0.3. The second correction is the loss factor due to radiation intensity observation through a medium other than that between the black-body and the imaging system (air). The loss factor was estimated from the spectral transmissivity of YAG, which was reported by the manufacturer. The effective wavelength seen by the imaging system is:

\[
\lambda_{\text{eff}} = A + \frac{B}{T}
\]  

(5.2)

For \( A \) and \( B \) above, and an expected temperature in the range from 750 °C to 1250 °C, \( \lambda_{\text{eff}} \) is in the range from 516 nm to 473 nm. For this wavelength, the transmissivity of the YAG is about 0.8. Hence, the loss factor is about 0.2.

After the calibration to obtain constants \( C, A \) and \( B \), the forward Sakuma-Hattori equation can be inverted algebraically to obtain the inverse Sakuma-Hattori equation, which has the form:

\[
T_{\text{true}} = \frac{c_2}{A \ln(C/F(T_{\text{true}}) + 1) - \frac{B}{A}}
\]  

(5.3)

where \( T_{\text{true}} \), in K, is the temperature of the chip underside for given radiation intensity \( F(T_{\text{true}}) \). Here, \( F(T_{\text{true}}) \), in counts, is taken after correcting for chip underside emissivity and through-the-YAG losses. If the uncorrected radiation intensity of the chip underside obtained from imaging
system is $S_{\text{measured}}$, then, knowing that the chip underside has emissivity $\varepsilon = 0.3$, and the through-the-YAG loss factor is $LF = 0.2$, then:

$$F(T_{\text{true}}) = \left( \frac{S_{\text{measured}}}{1 - LF} - (1 - \varepsilon).F(T_{\text{amb}}) \right)$$  \hspace{1cm} (5.4a)

and

$$F(T_{\text{amb}}) = \frac{C}{\exp\left(\frac{c_2}{AT_{\text{amb}} + B}\right) - 1}$$  \hspace{1cm} (5.4b)

where $T_{\text{amb}}$ (K) is the ambient temperature.

Equations 5.4 a and b can be used to transfer from radiation intensity measured at the chip-tool interface ($S_{\text{measured}}$) to the corrected radiation intensity ($F(T_{\text{true}})$); and Equation 5.3 can be used to transfer from corrected radiation intensity ($F(T_{\text{true}})$) to the true temperature of the chip underside ($T_{\text{true}}$).

**TABLE 5**

<table>
<thead>
<tr>
<th>BLACK-BODY CALIBRATION.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Black-Body Temperature (°C)</strong></td>
</tr>
<tr>
<td>700</td>
</tr>
<tr>
<td>900</td>
</tr>
<tr>
<td>982</td>
</tr>
<tr>
<td><strong>Hysteresis %</strong></td>
</tr>
</tbody>
</table>

After trial and error, it was determined that the machining would produce sufficient signal to noise ratio without saturation for an exposure time of 0.5 ms for cutting in air atmosphere, and 1 ms for all LN cooling cuts. These convenient exposure time values were utilized throughout the machining experiments.

The lathe feed was controlled by a feedback controller from Galil. The control software generated a voltage signal that was used to initiate lathe feed and to trigger image acquisition.
Therefore, lathe feed and image acquisition started simultaneously. In order to position the workpiece at a constant distance from the tool without making it contact the cutting edge, the workpiece was brought towards the tool until a precision-made plastic shim carefully inserted between the workpiece and the tool would no longer fall. After approaching the starting point, the lathe was given a step feed of 12.7 µm before checking if the shim had already been jammed. The shim thickness was 0.762 mm. Thus, this procedure ensured that the workpiece would be some 0.762 mm from the tool cutting edge. It was indeed 0.762 mm minus up to 6 times 12.7 µm, as the operator’s judgement influenced whether the shim would be taken as not-jammed or jammed. By following this positioning methodology, the tool remained unused up to actual machining for data acquisition. Each machining trial was set to 6 to 8 workpiece rotations, which implies total length of cut \((\pi \times \text{OD} \times \# \text{ of rotations}) = 0.38 \text{ m to 0.51 m.}\) At the end of the machining, the feed was reversed immediately after the last forward rotation.

![Graph](image)

Figure 7. Radiation intensity from black-body at temperatures from 700 °C to 982 °C. The solid orange curve is from the heating cycle and the dashed blue curve is from the cooling cycle.
A National Instruments data acquisition board was used in combination with LabVIEW to record the lathe/camera trigger signal, and to record machining force as measured by a Kistler piezo-based dynamometer (type 9272). The dynamometer channel used to measure the cutting force had a measuring range of -5 to 5 KN, resolution less than 0.01 N, linearity of ±1% of full scale output (FSO), hysteresis less than 1% FSO, stiffness of 0.4 KN/μm, and resonant frequency of 3.1 KHz. The dynamometer channel used to measure the thrust force had a measuring range of -5 to 20 KN, resolution less than 0.02 N, linearity of ±1% FSO, hysteresis less than 1% FSO, stiffness of 2 KN/μm, and resonant frequency of 6.3 KHz.

The upper side of the chips produced by the machining was imaged using an Alicona Infinite Focus 3D profilometer operated at minimum measurable height of 50 nm. The upper side is the side opposite to the one in contact with the tool, where shear localization at the primary shear plane usually results in well-defined ridges across the chip width. The tool rake face was imaged using a MicroXAM-100 3D profilometer operated at an optical resolution of 0.92 μm, to characterize tool wear (edge and/or crater wear) after the cutting.

For the cryogenic experiments, the setup was modified to allow LN to flow on the tool. The LN was initially transferred from a commercially available cylinder to a dewar, using an insulated flexible hose. The LN was transferred from the dewar to a funnel arrangement that was insulated with soft foam sheets. The funnel was filled and emptied twice to avoid boiling and stabilize the liquid. The volumetric flow rate of the LN flowing through the funnel was measured after the experiment and was found to be 75 ml/s. For the experiment with workpiece precooling, the LN flooded the workpiece, after applying it directly from the cylinder through its insulated flexible hose. Among the challenges during cryogenic experiments, fogging of the objective of ICCD camera and the mirror placed under the YAG tool were the most significant. To overcome
the challenge, a 3D printed cap was fixed to the tool holder which prevented the LN to flow on the camera objective. Gas nitrogen was directed on the mirror setup in the tool holder to prevent the mirror from fogging. The arrangement is as shown in Figure 8.

![Figure 8. Experimental setup for cryogenic experiments](image)

5.2 Experimental Conditions

For both the CW IN 625 and AM IN 625 workpieces the following procedure was performed. The rake angle was fixed at -5° and the relief angle at 5° by the tool holder design. The machining feed was kept constant at 50 µm/rev, while the machining velocity was changed from 1 m/s to 3 m/s, in increments of 1 m/s. There was also an attempt to run the experiment at 4 m/s and 5 m/s. Experiments with LN were performed at velocity of 2 m/s and feed of 50 µm/rev.

Two successful repeats were made for the condition with workpiece precooling and two repeats for experiments that had no workpiece precooling. For the same machining conditions, prior to the through-the-tool imaging using YAG tools, an attempt was made to capture the lateral face temperature of the chip tool interface. The tools used for these experiments were triangular carbide inserts. The rake angle and relief angles were 0° and 11°, respectively. These results were
used to evaluate the performance of the camera and to select the exposure time for through-the-tool imaging.

5.3 Data Processing

The software used to acquire images of the temperature field was programmed by Princeton Instruments, the manufacturer of the ICCD camera. The software provided an option for saving the images as tagged image file (TIF) format. A MATLAB graphical user interface (GUI) was programmed to carry out the post processing of the TIF images. A screenshot of the GUI is shown in Figure 9. The full code is given in Appendix A.

![MATLAB GUI for data processing](image)

Figure 9. MATLAB GUI programmed to process the thermal data collected from the camera.

The MATLAB GUI enabled the user to select the images that had temperature data, rotate them, and crop them to zoom into the final region of interest, where the radiation intensity was relatively uniform across the width of the cut. The images from a sample of the background noise for the exposure time values commonly used throughout this study were built in the code. The code subtracts, pixel-by-pixel, the mean background noise image from each of the images to be analyzed. The background noise images are pulled from the built-in database after the user types
the exposure time that was used. The net radiation intensity is converted, pixel by pixel, by the code as per Equations 5.3 and 5.4 a and b.

The GUI setup processs comprising the selection, rotation, cropping, background noise removal, and conversion to temperature is based on the image that the user has asked the GUI to display. This image is from one of the frames in the time series available in the TIF file. The setup choices are applied to all the other frames in the time series. The code then computes the mean temperature across the width of the cut as a function of distance along the tool rake face. If the crop region is carefully selected so its left side coincides with the tool cutting edge, then the distance along the rake face is indeed the distance from the cutting edge. The left side of the screenshot in Figure 9 shows a sample image that is being used to complete the GUI setup process. The right side of this screenshot shows the mean temperature across the width of the cut vs. distance from the cutting edge, for the sample image displayed on the left. The error bars on the plot are the standard deviations calculated from the temperature values across the width of the cut. The code repeats the calculation of mean temperature across the width of the cut vs. distance from the cutting edge for all selected images in the TIF file; and then calculates, pixel-by-pixel, the mean temperature field for the time spanned by the selected images. The error in each time averaged value is calculated using the formula for the standard deviation of the mean. The input to the GUI is the time series in the TIF file that corresponds to the radiation intensity field obtained from a given cutting experiment. The outputs from the GUI are the mean temperature across the width of the cut vs. distance from the cutting edge for each image in the time series, and the time averaged – or equivalent – mean temperature vs. distance from the cutting edge. The time series corresponding to the mean temperature across the width of the cut vs. distance from the cutting edge, or mean temperature profiles, for all the cutting experiments are shown in Figure 12, Figure
The time averaged temperature profiles for all the cutting experiments are shown in Figure 14, Figure 15 and Figure 31.

A similar MATLAB GUI was programmed for the processing of the force data. The force data (time series) recorded by LabVIEW is imported to this GUI, along with the time series formed by the peak temperature from the images that were analyzed by the MATLAB GUI for temperature processing. The code plots the cutting and thrust forces, and the peak temperature as functions of time. The different signals are synchronized as explained in the Experimental Configuration. Figure 10 shows a screenshot of the MATLAB GUI that was used to produce the force and peak temperature time series. The full code is given in Appendix B.
6.1 Machinability of AM IN 625 vs. CW IN 625

6.1.1 Temperature field

An example image (frame) of the radiation intensity field recorded by the camera appears in Figure 11. This image shows the full field on the tool rake face and a reflection of it from the tool flank face. The dashed line separating the two symmetric regions on the image corresponds to the tool cutting edge. The side to the right of this line corresponds to the direct image of the rake face. The side to the left of this line corresponds to the reflection of the rake face on the flank face. The chip flows perpendicular to the dashed line towards its right side. Note that the example image captured nearly the full width of the workpiece, and that besides drops in radiation intensity near both workpiece sides, the field, along this width, is uniform. With increasing distance from the cutting edge along the chip flow direction, however, the radiation intensity first rises and then drops. Thus, the point of maximum radiation intensity along the tool rake face is at a finite distance from the cutting edge. This behavior was observed in all machining trials both in an air atmosphere and in a LN atmosphere. The uniform field shape across most of the chip width makes it possible to extract mean radiation intensity profiles. That is, radiation intensity vs. distance from the cutting edge. The calculation involves averaging the intensities at a given distance from the tool cutting edge over a band at the center of the cut. Radiation intensity can then be transferred into true temperature with the use of equations (3.3) and (3.4a and b). This calculation was performed for bands that were 100 to 150 µm wide (10% to 15% of the width of the cut).

Figure 12 and Figure 13 show the temperature profiles from each image (frame) from the very start of the machining trial to the very end, for CW IN 625 and AM IN 625, respectively. The
temperature is the vertical axis. The other two axes are distance from the cutting edge and time along the cut. Note that the cutting edge is at the origin, but time zero is not. Time zero is the start of lathe motion and image acquisition. This choice of time zero enabled synchronization between temperature and machining force time histories, as will be shown in the forthcoming. From Figure 12 and Figure 13, it is clear that the temperature profile reached a steady state within one or two rotations of the workpiece (< 80 ms).

![A) Magnified schematic of the cutting arrangement from the point of view of the camera. B) A sample image of radiation intensity as measured by the camera.](image)

To extract meaningful statistics from the temperature profiles, first the mean image from the period identified as steady-state machining is calculated by averaging true temperatures, pixel by pixel, across the frames that fall in this period. The standard deviation calculated also from the true temperatures, pixel by pixel, is recorded along with the corresponding mean true temperatures. The band corresponding to the middle of the workpiece width is then identified, and the true temperature values that are at a given distance from the tool cutting edge are again averaged. The standard deviation characterizing the measurement error is taken as the standard deviation of the mean, which is computed from both the true temperatures at a given pixel across the images, and the true temperatures at a given distance from the cutting edge across the width of the workpiece.
The mean temperature profiles and the corresponding measurement errors (half a bar being one standard deviation) for all machining trials are shown in Figure 14 and Figure 15. From these mean temperature profiles, the peak temperature and its corresponding error, and the distance from the cutting edge to the peak temperature are extracted and summarized in Table 6. The error in the distance from the cutting edge to the peak temperature is taken as the width of one pixel (10.24 μm).

There are important differences between the tool rake face temperature distribution when machining AM IN 625 relative to the temperature distribution when machining CW IN 625. Figure 16 and Figure 17 indicate that, for a given machining velocity, the AM material produces peak temperature higher than the CW material. The larger temperature observations are consistent with AM IN 625 being stronger than CW IN 625.
Figure 12. Evolution of the temperature profiles after cutting **CW IN 625**. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2.
Figure 13. Evolution of the temperature profiles after cutting AM IN 625. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2.
Figure 14. Average temperature profile as a function of time for several machining velocities. **Work material: CW IN 625.** Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, cut width = 0.8 mm.
Figure 15. Average temperature profile as a function of time for several machining velocities. 
**Work material: AM IN 625.** Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, cut width = 0.8 mm.
Figure 16. Average temperature at the tool rake face as a function of distance from the cutting edge for a range of machining velocity between 1 m/s and 3 m/s. **Work material:** CW IN 625. Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, cut width = 0.8 mm.

Figure 17. Average temperature at the tool rake face as a function of distance from the cutting edge for a range of machining velocity between 1 m/s and 3 m/s. **Work material:** AM IN 625. Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, cut width = 0.8 mm.
6.1.2 Machining forces

From Figure 18 and Figure 19 it is also clear that both the cutting and the thrust force reach steady state within one or two rotations of the workpiece. During the initial rotations, when the temperature profile is higher than those from the rest of the cut, so is the machining force. This spike in temperature and force is probably due to the step left on the workpiece after retracting it. The spikes can be avoided by finishing the workpiece flat before the next machining trial, but for the data that is available, the finishing procedure was not performed. The first two rotations were not included in final statistics. The dashed vertical lines printed on the plots in Figure 18 and Figure 19 show the period from which the statistics (mean temperature and machining force) were calculated. The coefficient of friction, calculated using equation (3.5), and equations (3.3) and (3.4), is plotted in Figure 20 and Figure 21. Summary statistics are shown in Table 6.

| TABLE 6. |
| SUMMARY STATISTICS COMPARING CW IN 625 AND AM IN 625 |

<table>
<thead>
<tr>
<th>Material/Machining velocity</th>
<th>Peak Temperature (°C)</th>
<th>Distance from the cutting edge to the peak temperature (µm)</th>
<th>Cutting Force (N)</th>
<th>Thrust Force (N)</th>
<th>Coefficient of friction</th>
</tr>
</thead>
<tbody>
<tr>
<td>CW IN 625/ 1 m/s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>943 ± 7</td>
<td>122.9 ± 10.2</td>
<td>172 ± 7</td>
<td>114 ± 6</td>
<td>0.54 ± 0.03</td>
</tr>
<tr>
<td>Trial 2</td>
<td>943 ± 2</td>
<td>122.9 ± 10.2</td>
<td>161 ± 11</td>
<td>112 ± 8</td>
<td>0.58 ± 0.04</td>
</tr>
<tr>
<td>Mean</td>
<td>943</td>
<td>122.9</td>
<td>167</td>
<td>113</td>
<td>0.56</td>
</tr>
<tr>
<td>AM IN 625/ 1 m/s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>973 ± 2</td>
<td>133.1 ± 10.2</td>
<td>165 ± 13</td>
<td>108 ± 8</td>
<td>0.54 ± 0.05</td>
</tr>
<tr>
<td>Trial 2</td>
<td>944 ± 2</td>
<td>133.1 ± 10.2</td>
<td>154 ± 12</td>
<td>107 ± 12</td>
<td>0.58 ± 0.06</td>
</tr>
<tr>
<td>Mean</td>
<td>959</td>
<td>133.1</td>
<td>160</td>
<td>108</td>
<td>0.56</td>
</tr>
<tr>
<td>CW IN 625/ 2 m/s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>1048 ± 1</td>
<td>92.2 ± 10.2</td>
<td>151 ± 11</td>
<td>103 ± 12</td>
<td>0.57 ± 0.07</td>
</tr>
<tr>
<td>Trial 2</td>
<td>1042 ± 5</td>
<td>92.2 ± 10.2</td>
<td>141 ± 34</td>
<td>101 ± 17</td>
<td>0.6 ± 0.2</td>
</tr>
<tr>
<td>Mean</td>
<td>1045</td>
<td>92.2</td>
<td>146</td>
<td>102</td>
<td>0.6</td>
</tr>
<tr>
<td>AM IN 625/ 2 m/s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>1112 ± 3</td>
<td>112.6 ± 10.2</td>
<td>157 ± 22</td>
<td>89 ± 22</td>
<td>0.5 ± 0.1</td>
</tr>
<tr>
<td>Trial 2</td>
<td>1079 ± 1</td>
<td>112.6 ± 10.2</td>
<td>150 ± 15</td>
<td>87 ± 12</td>
<td>0.47 ± 0.08</td>
</tr>
<tr>
<td>Mean</td>
<td>1096</td>
<td>112.6</td>
<td>154</td>
<td>88</td>
<td>0.5</td>
</tr>
<tr>
<td>CW IN 625/ 3 m/s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>1144 ± 1</td>
<td>81.9 ± 10.2</td>
<td>156 ± 14</td>
<td>91 ± 24</td>
<td>0.5 ± 0.1</td>
</tr>
<tr>
<td>Trial 2</td>
<td>1059 ± 1</td>
<td>81.9 ± 10.2</td>
<td>158 ± 20</td>
<td>101 ± 32</td>
<td>0.5 ± 0.2</td>
</tr>
<tr>
<td>Mean</td>
<td>1102</td>
<td>81.9</td>
<td>157</td>
<td>96</td>
<td>0.5</td>
</tr>
<tr>
<td>AM IN 625/ 3 m/s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>1194 ± 3</td>
<td>71.7 ± 10.2</td>
<td>214 ± 29</td>
<td>143 ± 36</td>
<td>0.6 ± 0.2</td>
</tr>
<tr>
<td>Trial 2</td>
<td>1147 ± 3</td>
<td>71.7 ± 10.2</td>
<td>161 ± 14</td>
<td>150 ± 18</td>
<td>0.8 ± 0.1</td>
</tr>
<tr>
<td>Mean</td>
<td>1171</td>
<td>71.7</td>
<td>188</td>
<td>147</td>
<td>0.7</td>
</tr>
</tbody>
</table>
Since the peak temperature at the tool rake face when cutting AM IN 625 is larger than when cutting CW IN 625 (Table 6), so should be the machining force. However, the cutting and thrust forces, and the coefficient of friction shown in Table 6, for both the AM and the CW work materials are not statistically different from each other. The large error in force measurement is probably due to process variability arising from excessive tool wear. As will be seen in the following section, the YAG tools experienced significant wear, especially on the lateral ends of the chip-tool contact (workpiece sides). It would be beneficial to measure the machining force while cutting with tools that do not wear. Coated tungsten carbide tools may be more appropriate than YAG tools for machining force measurement.
Figure 18. Evolution of the peak temperature and the machining force after cutting CW IN 625.
A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2. The force labeled “X force” is the force perpendicular to the cutting and thrust forces. It is essentially “zero” as expected from orthogonal cutting.
Figure 19. Evolution of the peak temperature and the machining force after cutting AM IN 625.
A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2. The force labeled “X force” is the force perpendicular to the cutting and thrust forces. It is essentially “zero” as expected from orthogonal cutting.
Figure 20. Coefficient of friction as the function of time through the cut for a range of machining velocity between 1 m/s and 3 m/s. **Work material:** CW IN 625. Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, cut width = 0.8 mm.
Figure 21. Coefficient of friction as the function of time through the cut for a range of machining velocity between 1 m/s and 3 m/s. **Work material:** AM IN 625. Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 μm/rev, cut width = 0.8 mm.
6.1.3 Tool Wear

Figure 22 and Figure 23 show images of the rake face of the tool, for the cutting of AM IN 625 and CW IN 625, respectively. Figure 24 and Figure 25 show the wear patterns from the tool cutting edge and rake face, also for the cutting of AM IN 625 and CW IN 625, respectively. The plots in Figure 26 are mean wear profiles extracted after averaging the wear patterns across the whole width of the cut. The area above the mean wear profile is the volumetric loss of cutting tool material, per unit width. These width-normalized volumetric losses are summarized in Table 7.

![Image of a tool and its wear patterns](image)

**TABLE 7**

**SUMMARY OF TOOL WEAR IN TERMS OF VOLUME LOSS PER UNIT WIDTH OF THE TOOL**

<table>
<thead>
<tr>
<th>Material/Machining velocity</th>
<th>Volume loss per unit width (µm²)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CW IN 625/ 1 m/s</td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>228</td>
</tr>
<tr>
<td>Trial 2</td>
<td>61</td>
</tr>
<tr>
<td>AM IN 625/ 1 m/s</td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>355</td>
</tr>
<tr>
<td>Trial 2</td>
<td>514</td>
</tr>
<tr>
<td>CW IN 625/ 2 m/s</td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>180</td>
</tr>
<tr>
<td>Trial 2</td>
<td>112</td>
</tr>
<tr>
<td>AM IN 625/ 2 m/s</td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>758</td>
</tr>
<tr>
<td>Trial 2</td>
<td>471</td>
</tr>
<tr>
<td>CW IN 625/ 3 m/s</td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>N/A</td>
</tr>
<tr>
<td>Trial 2</td>
<td>63</td>
</tr>
<tr>
<td>AM IN 625/ 3 m/s</td>
<td></td>
</tr>
<tr>
<td>Trial 1</td>
<td>788</td>
</tr>
<tr>
<td>Trial 2</td>
<td>1082</td>
</tr>
</tbody>
</table>

From the wear patterns and the volumetric losses per unit width, it is clear AM IN 625 produces more wear than CW IN 625. It is also clear that the wear is mainly confined to the lateral sides of the chip-tool contact region, but some edge wear is present across the contact width. This result suggests that the AM IN 625 is more difficult to machine than CW IN 625. The suggestion is consistent with the strength difference between AM IN 625 and CW IN 625 – the strength of
the AM material being larger than the strength of the CW material. An inspection of Figure 22 and 24 also reveals that material transfer from the CW IN 625 chip to the tool rake face is confined to the outer edges of the chip-tool contact; while the transfer from the AM IN 625 chip to the tool rake face is quite uniform throughout the contact region. The differences in this material transfer behavior may be explained by differences in oxygen content in the two materials. Oxygen from air may react with the oxygen-poor CW material; but the reaction, if really occurring, is unlikely to penetrate the intimate region of chip-tool contact. The same reaction may be occurring farther into the chip-tool contact region for the case of the oxygen-rich AM material. The oxygen content of both work materials needs to be determined before drawing definite conclusions explaining the differences in chip-to-tool material transfer behavior.
Figure 22. Tool rake face after cutting **CW IN 625**. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2.
Figure 23. Tool rake face after cutting **AM IN 625**. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2.
Figure 24. Surface profile of rake face after cutting CW IN 625. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2.
Figure 25. Surface profile of rake face after cutting **AM IN 625**. A) Machining velocity = 1 m/s; left: trial 1, right: trial 2. B) Machining velocity = 2 m/s; left: trial 1, right: trial 2. C) Machining velocity = 3 m/s; left: trial 1, right: trial 2.
Figure 26. Tool rake face wear profile after the machining. A), B), and C) corresponds to cuts with CW IN 625 at machining velocity = 1 m/s, 2m/s and 3 m/s resp. D), E), and F) corresponds to cuts with AM IN 625 at machining velocity of 1 m/s, 2m/s, and 3 m/s resp.
6.1.4 Chip Morphology

A notable difference between AM and CW materials is observed in the morphology of the chips formed. AM chips show a rather random serration pattern (Figure 27 b) that suggests that at any given time the material deforms along multiple distinct bands along the width of the workpiece, which is characteristic of inhomogeneous shear along weaker regions of the material. In contrast, the wrought chips show typical, well-defined, discrete adiabatic shear bands that run through the entire width of the chip (Figure 27 a).

Figure 27. a) Typical CW IN 625 chips, b) Typical AM IN 625 chips. V = machining velocity.
6.2 Cryogenic Machining

The temperature field in the middle portion of the chip-tool contact is shown in Figure 28, for the case of cutting in an air atmosphere and cutting under the action of the LN jet. Note that the temperature field is given as a function of time into the cut. For a given time (frame #), the field is plotted as temperature vs. distance from the cutting edge. Figure 28 A corresponds to the cuts made in the air atmosphere, and Figure 28 B and C show the cuts made under the LN jet. In the figures, the total time corresponds to the total duration of the cut, which was equivalent to 4 or 5 rotations for the cuts in air, and 8 to 9 rotations for the cuts with LN. The inter-frame time for the cuts in air was 39.42 ms, while that for the cuts with LN was 39.92 ms. Therefore, the data was gathered at essentially the same frame rate for all four cuts. Figure 29 shows the peak temperature from the profiles in Figure 28, along with the cutting and thrust force traces. Figure 29 A corresponds to the cuts made in the air atmosphere, and Figure 29 B and C show the cuts made under the LN jet. The time-averaged profiles for the air and LN cases, compiled from Figure 28, are shown in Figure 31. These profiles (and the peak temperatures in Figure 29) suggest that the peak temperature at the chip-tool interface drops from ~1050 °C in air, to about ~950 °C with LN. For machining in an air atmosphere, the peak temperature results in the highest value (~ 1050 ± 5 °C). For machining in a LN atmosphere without pre-cooling of the workpiece, the peak temperature is intermediate (~ 1000 ± 5 °C). For machining in a LN atmosphere with pre-cooling of the workpiece, the peak temperature results in the lowest value (~ 950 ± 5 °C). The machining forces are observed to be higher in case of the cuts with LN (Figure 29).
Figure 28. Evolution of the temperature profiles. A) Machining in air atmosphere; left: trial 1, right: trial 2. B) Machining in LN atmosphere without workpiece pre-cooling; left: trial 1, right: trial 2. C) Machining in LN atmosphere with workpiece pre-cooling; left: trial 1, right: trial 2.
Figure 29. Evolution of the peak temperature and the machining force. A) Machining in air atmosphere; left: trial 1, right: trial 2. B) Machining in LN atmosphere without workpiece pre-cooling; left: trial 1, right: trial 2. C) Machining in LN atmosphere with workpiece pre-cooling. The force labeled “X force” is the force perpendicular to the cutting and thrust forces. It is essentially “zero” as expected from orthogonal cutting.
Figure 30. Wear patterns on the cutting edge and the tool rake face after the machining. A) Machining in air atmosphere. B) Machining in LN atmosphere with workpiece pre-cooling.
Table 8 shows that the tool wear is minimal in case of cuts in air atmosphere. For cuts in air, the mean volumetric loss of the tool material per unit width was 146 µm$^2$. During the cryogenic machining, the volumetric loss was 601 µm$^2$ for cuts without workpiece precooling, and 641 µm$^2$ for cuts with workpiece precooling. It should be noted that the cuts with LN were for 8 rotations, whereas the cuts in air were for 6 rotations. The longer cuts could be the culprit regarding the increased wear with LN, but this increased wear may also be the result of increased machining forces. Although there was a higher tool wear during the cryogenic cuts, the temperature at the rake face dropped when compared to that from the cutting in air.

As a final summary of the most important findings from the cryogenic machining efforts, Figure 31 highlights the mean temperature profile at the rake face as a function of distance from the cutting edge, and Table 9 gives the statistics for peak temperature, distance to the peak temperature from the cutting edge, and machining forces.
Figure 31. Average temperature at the tool rake face as a function of distance from the cutting edge under air atmosphere and LN cooling condition. **Work material: CW IN 625.** Machining conditions: tool = YAG, tool rake angle = -5°, relief angle = 5°, feed = 50 µm/rev, V = 2 m/s, cut width = 0.8 mm.

**TABLE 9**  
SUMMARY STATISTICS COMPARING AIR ATMOSPHERE CUTS WITH LIQUID NITROGEN CUTS.

<table>
<thead>
<tr>
<th>Atmosphere</th>
<th>Peak Temperature (°C)</th>
<th>Distance from the cutting edge to the peak temperature (µm)</th>
<th>Cutting Force (N)</th>
<th>Thrust Force (N)</th>
<th>Coefficient of friction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Air</td>
<td>Trial 1: 1048 ± 1</td>
<td>92.2 ± 10.2</td>
<td>168 ± 12</td>
<td>114 ± 13</td>
<td>0.57 ± 0.07</td>
</tr>
<tr>
<td></td>
<td>Trial 2: 1029 ± 5</td>
<td>92.2 ± 10.2</td>
<td>157 ± 38</td>
<td>112 ± 19</td>
<td>0.6 ± 0.2</td>
</tr>
<tr>
<td></td>
<td>Mean: 1039</td>
<td>92.2</td>
<td>163</td>
<td>113</td>
<td>0.6</td>
</tr>
<tr>
<td>Liquid Nitrogen - no precooling</td>
<td>Trial 1: 981 ± 1</td>
<td>102.4 ± 10.2</td>
<td>255 ± 13</td>
<td>161 ± 14</td>
<td>0.52 ± 0.06</td>
</tr>
<tr>
<td></td>
<td>Trial 2: 1012 ± 2</td>
<td>102.4 ± 10.2</td>
<td>258 ± 17</td>
<td>193 ± 12</td>
<td>0.62 ± 0.05</td>
</tr>
<tr>
<td></td>
<td>Mean: 997</td>
<td>102.4</td>
<td>257</td>
<td>177</td>
<td>0.57</td>
</tr>
<tr>
<td>Liquid Nitrogen - precooling</td>
<td>Trial 1: 972 ± 2</td>
<td>102.4 ± 10.2</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>Trial 2: 942 ± 2</td>
<td>102.4 ± 10.2</td>
<td>209 ± 14</td>
<td>121 ± 19</td>
<td>0.47 ± 0.09</td>
</tr>
<tr>
<td></td>
<td>Mean: 957</td>
<td>102.4</td>
<td>209</td>
<td>121</td>
<td>0.47</td>
</tr>
</tbody>
</table>
CHAPTER 7
DISCUSSION

There are significant differences between the machinability of AM IN 625 and CW IN 625. The differences are in maximum temperature at the chip-tool interface, tool wear, and chip formation mechanisms.

As it can be seen from Figure 32 and Table 6, the peak temperature at the chip-tool interface during the machining of the AM material is about 16, 51, and 69 °C higher than that of the CW material, for machining velocities of 1, 2 and 3 m/s, respectively. The distance from the cutting edge to the peak temperature for AM and CW materials are nearly identical. However, for both materials, this distance decreases with increase in machining velocity.

![Average temperature profile](image.png)

Figure 32. Average temperature profile while machining CW IN 625 and AM IN 625 for machining velocities 1 m/s, 2 m/s, and 3 m/s.
Figure 33. Temperature profile and tool wear patterns superimposed on rake face images A) Corresponds to CW IN 625 B) Corresponds to AM IN 625. V = Machining velocity.
As it can be seen from Figure 33 and Table 7, the volume loss at the tool cutting edge and the rake face during the machining of the AM material is about 3, 4, and 15 times that of the CW material, for machining velocities of 1, 2 and 3 m/s, respectively. Most of this volume loss occurs at the sides of the chip-tool contact where the lateral edges of the chip contact the tool (dark craters in Figure 33). After machining the AM material, the tool cutting edge was significantly worn across the full width of chip-tool contact, for machining velocity = 1 and 3 m/s. However, for machining velocity = 2 m/s, the tool cutting edges after machining both the AM and the CW materials were nearly intact (Figure 33).

The wear makes temperature and force comparisons rather difficult, but since the tool cutting edges from both the AM and the CW materials at machining velocity = 2 m/s remained nearly intact, and the lateral crater wear presumably has a negligible effect on temperature, temperature comparisons at this machining velocity are possible. The assumption being made is that since the tool temperature is measured between the lateral craters on the tool rake face, it will be controlled by the contact at this region where insignificant wear is observed. Based on the observations at this machining velocity, the tool temperature from the cutting of the AM is about 50 ºC higher than that from the cutting of the CW.

As can be seen from Table 6, the mean cutting and thrust forces, and the coefficient of friction from the machining of the AM and CW materials are all within the data scatter. It was expected that tool temperature and machining force would be correlated, and that the higher ultimate tensile strength and machining temperatures measured for the AM material should also have resulted in a higher machining force. As per the measurements, it cannot be concluded that the force from the machining of the AM material is higher than that of the CW material. If there
was no tool wear, it is likely that the force measured while machining the AM material would be higher than that measured while machining the CW material.

The parasitic force from the significant lateral crater wear may be a strong component of the total force. Therefore, the assumption that the lateral craters do not affect temperature cannot be extended to force. Temperature and wear are intrinsically related to one another, therefore, since there was tool wear throughout the experiments, it is not possible to fully isolate the effect of ultimate tensile strength on temperature. Since the total volume loss for AM was, for all machining trials, higher than the total volume loss for CW, it can be concluded that the AM material is indeed more difficult to machine than the CW material. There is significant material transfer from the chip to the tool rake face for AM, but not for CW (Figure 33). The transfer may be due to higher oxygen content in the AM material than in the CW material.

As can be seen in Figure 27, the AM IN 625 chips at all machining velocities from 2 to 4 m/s are characterized by a rather non-uniform segmentation across their width (direction parallel to the workpiece width); whereas the CW IN 625 chips are all indicative of adiabatic shear banding, and all show uniform segmentation across their width. The non-uniform segmentation is likely to be due to hard second phase particles present in the rapidly solidified AM material, and it is consistent with its lower elongation at breakage.

It is noteworthy to mention that the direct tool rake face temperature measurements offer the most accurate estimates for the temperature at the chip-tool interface. Measurements of the temperature at the tool lateral sides, even if performed right up to the immediate vicinity of the tool rake face, are plagued with many challenges including difficulties in keeping lateral chip spread from blocking the field of view and severe uncertainty in the emissivity of the tool material, which is prone to oxidation during the cutting. In fact, measurements of the temperature at the tool
and workpiece lateral sides were performed during the initial phases of the work presented herein, but excluded from this thesis due to significant observation uncertainty.

There is a dependency of the temperature estimates on the emissivity of the workpiece used in the study. The emissivity value for the material used for all the calculations was taken from the work done by Kobayashi et al. [26]. The value of the emissivity of unoxidized, polished IN 625 was 0.3. The chip underside is likely to be “unoxidized and polished”, as it is in intimate contact with a very smooth face – the tool rake face; and hence an emissivity of 0.3 is likely to be the real emissivity. However, changes in temperature estimates as a function of emissivity value, as per equations (5.3), and (5.4 a and b), are summarized in Table 10, for a range of emissivity values from 0.25 to 0.35. The data in the table shows that for an emissivity error of 17% (from 0.3 to either 0.25 or 0.35), the temperature error is no more than about 1%. The “logarithmic” nature of the relationship between temperature and radiation intensity, as expressed by the Sakuma-Hattori equation, makes the temperature fairly unsensitive to emissivity variations.

TABLE 10
TEMPERATURE VS EMISSIVITY FOR A RANGE OF RADIATION INTENSITY

<table>
<thead>
<tr>
<th>Emissivity</th>
<th>Counts/ms</th>
<th>Temperature (°C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>480</td>
<td>810</td>
<td>0.25</td>
</tr>
<tr>
<td>2700</td>
<td>912</td>
<td>0.26</td>
</tr>
<tr>
<td>12200</td>
<td>1013</td>
<td>0.27</td>
</tr>
<tr>
<td>46200</td>
<td>1115</td>
<td>0.28</td>
</tr>
<tr>
<td>480</td>
<td>808</td>
<td>0.29</td>
</tr>
<tr>
<td>2700</td>
<td>910</td>
<td>0.30</td>
</tr>
<tr>
<td>12200</td>
<td>1008</td>
<td>0.31</td>
</tr>
<tr>
<td>46200</td>
<td>1106</td>
<td>0.32</td>
</tr>
<tr>
<td>480</td>
<td>806</td>
<td>0.33</td>
</tr>
<tr>
<td>2700</td>
<td>907</td>
<td>0.34</td>
</tr>
<tr>
<td>12200</td>
<td>1005</td>
<td>0.35</td>
</tr>
<tr>
<td>46200</td>
<td>1103</td>
<td></td>
</tr>
<tr>
<td>480</td>
<td>804</td>
<td></td>
</tr>
<tr>
<td>2700</td>
<td>905</td>
<td></td>
</tr>
<tr>
<td>12200</td>
<td>1003</td>
<td></td>
</tr>
<tr>
<td>46200</td>
<td>1100</td>
<td></td>
</tr>
<tr>
<td>480</td>
<td>802</td>
<td></td>
</tr>
<tr>
<td>2700</td>
<td>902</td>
<td></td>
</tr>
<tr>
<td>12200</td>
<td>1000</td>
<td></td>
</tr>
<tr>
<td>46200</td>
<td>1100</td>
<td></td>
</tr>
<tr>
<td>480</td>
<td>800</td>
<td></td>
</tr>
<tr>
<td>2700</td>
<td>900</td>
<td></td>
</tr>
<tr>
<td>12200</td>
<td>1000</td>
<td></td>
</tr>
<tr>
<td>46200</td>
<td>1100</td>
<td></td>
</tr>
<tr>
<td>480</td>
<td>799</td>
<td></td>
</tr>
<tr>
<td>2700</td>
<td>899</td>
<td></td>
</tr>
<tr>
<td>12200</td>
<td>998</td>
<td></td>
</tr>
<tr>
<td>46200</td>
<td>1097</td>
<td></td>
</tr>
<tr>
<td>480</td>
<td>798</td>
<td></td>
</tr>
<tr>
<td>2700</td>
<td>897</td>
<td></td>
</tr>
<tr>
<td>12200</td>
<td>996</td>
<td></td>
</tr>
<tr>
<td>46200</td>
<td>1095</td>
<td></td>
</tr>
<tr>
<td>480</td>
<td>796</td>
<td></td>
</tr>
<tr>
<td>2700</td>
<td>895</td>
<td></td>
</tr>
<tr>
<td>12200</td>
<td>993</td>
<td></td>
</tr>
<tr>
<td>46200</td>
<td>1093</td>
<td></td>
</tr>
<tr>
<td>480</td>
<td>794</td>
<td></td>
</tr>
<tr>
<td>2700</td>
<td>893</td>
<td></td>
</tr>
<tr>
<td>12200</td>
<td>992</td>
<td></td>
</tr>
<tr>
<td>46200</td>
<td>1090</td>
<td></td>
</tr>
<tr>
<td>480</td>
<td>792</td>
<td></td>
</tr>
<tr>
<td>2700</td>
<td>891</td>
<td></td>
</tr>
<tr>
<td>12200</td>
<td>990</td>
<td></td>
</tr>
<tr>
<td>46200</td>
<td>1088</td>
<td></td>
</tr>
</tbody>
</table>

As observed from Figure 31 and Table 9, after the cryogenic machining of CW IN 625 it was found that: (1) Precooling the workpiece with LN, in combination with a continuous application on the tool rake face of a jet of LN, resulted in the lowest peak temperature (~957±2°C); (2) Continuous application on the tool rake face of a jet of LN without precooling the
workpiece resulted in an intermediate peak temperature of ~ 997±2 °C; (3) Machining in air atmosphere resulted in the highest peak temperature (~ 1039±5 °C). Given that the temperature ranges, within plus or minus one standard deviation, do not overlap, the temperature differences are significant. From Figure 29 and Table 9, the cutting and thrust forces were highest for “precooling plus continuous application of LN”, and lowest for machining in air atmosphere. This result indicates that the strength of IN 625 may be significantly sensitive to temperature drop below the ambient level. Machining with LN produced material transfer to the tool rake face (Figure 30 A). This transfer was insignificant when machining in air atmosphere (Figure 30 B). The transfer may be due to the formation of a compound between one or more of the constituents of IN 625 and nitrogen.
CHAPTER 8
CONCLUSIONS

From the analysis presented herein, it can be concluded that:

1) The peak temperature at the chip-tool interface during the machining of AM IN 625 produced by LPBF is about 16, 51, and 69 °C higher than that of the CW material of similar composition, for machining velocities of 1, 2 and 3 m/s, respectively.

2) The distance from the cutting edge to the peak temperature for AM and CW materials are nearly identical. However, this distance decreases with increase in machining velocity.

3) For identical machining conditions, tool wear from machining the AM material is more severe than from machining the CW counterpart. Tool wear increases with machining velocity. For the YAG tool material, the tool wear for AM material, as measured from total volume loss, after about ½ m of cut length, is 3, 4, and 14 times that for CW material.

4) There is significant material transfer from the chip to the tool rake face for AM, but not for CW. The transfer may be due to higher oxygen content in the AM material than in the CW material.

5) The AM material produces chips that show non-uniform segmentation across the chip width; whereas the CW material produces chips that show uniform segmentation. The findings suggest that adiabatic shear banding may be inoperative during the machining of the AM material, probably due to randomly spaced second phase particles dominating damage. The CW material shows signs of adiabatic shear banding during machining.

6) The higher temperature and wear, and the non-uniform segmentation in the machining of the AM material are consistent with the material’s higher strength and lower ductility, when compared to the CW counterpart.
7) Differences in machining force for the AM and CW materials could not be resolved, due to tool wear, especially at the lateral sides of the chip-tool contact.

8) After the cryogenic machining of CW IN 625 it was found that: (1) Precooling the workpiece with LN, in combination with a continuous application on the tool rake face of a jet of LN, resulted in the lowest peak temperature (~ 957 °C); (2) Continuous application on the tool rake face of a jet of LN without precooling the workpiece resulted in an intermediate peak temperature of ~ 997 °C; (3) Machining in air atmosphere resulted in the highest peak temperature (~ 1039 °C).

9) The cutting and thrust forces were highest for “precooling plus continuous application of LN”, and lowest for machining in air atmosphere. This result indicates that the strength of IN 625 may be significantly sensitive to temperature drop below the ambient level.

10) Machining with LN produced material transfer to the tool rake face. This transfer was insignificant when machining in air atmosphere. The transfer may be due to the formation of a compound between one or more of the constituents of IN 625 and nitrogen.
CHAPTER 9

FUTURE WORK

1) Both force and temperature should be measured while cutting with tools that undergo insignificant wear, so that the effects of material behavior alone can be resolved.

2) There is a need to expand the ranges of strain and strain rate, to derive constitutive models for the material.

3) To decrease the uncertainty of the temperature measurement, the emissivity of the chip underside must be determined, and the effects of the material transfer onto the tool on the emissivity considered. A non-uniformity correction must be performed and applied to the images.

4) Mitigation of YAG tool wear is required. To achieve wear resistance, it may be needed to coat the tool with a thin layer of an industrial grade tool material, or even diamond.

5) Alternative methods for LN delivery should be evaluated. Among these, delivery through microfluidic channels embedded in the cutting tool may be implemented to cool the region of the tool in the immediate vicinity of the rake face and the chip underside.
REFERENCES
REFERENCES


APPENDICES
APPENDIX A

TEMPERATURE DATA PROCESSING MATLAB CODE

function varargout = TemperatureGUI(varargin)
% TEMPERATUREGUI MATLAB code for TemperatureGUI.fig
% TEMPERATUREGUI, by itself, creates a new TEMPERATUREGUI or raises the
% existing
% singleton*.
% % H = TEMPERATUREGUI returns the handle to a new TEMPERATUREGUI or the
% handle to
% the existing singleton*.
% % TEMPERATUREGUI('CALLBACK',hObject,eventData,handles,...) calls the
local
% function named CALLBACK in TEMPERATUREGUI.M with the given input
arguments.
% % TEMPERATUREGUI('Property','Value',...) creates a new TEMPERATUREGUI or
raises the
% existing singleton*. Starting from the left, property value pairs are
% applied to the GUI before TemperatureGUI_OpeningFcn gets called. An
% unrecognized property name or invalid value makes property application
% stop. All inputs are passed to TemperatureGUI_OpeningFcn via
varargin.
% % See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
% instance to run (singleton)".
% % See also: GUIDE, GUIDATA, GUIHANDLES
% Edit the above text to modify the response to help TemperatureGUI
% Last Modified by GUIDE v2.5 30-Nov-2017 11:30:38
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @TemperatureGUI_OpeningFcn, ...
    'gui_OutputFcn', @TemperatureGUI_OutputFcn, ...
    'gui_LayoutFcn', [], ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% *** Executes just before TemperatureGUI is made visible.
function TemperatureGUI_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to TemperatureGUI (see VARARGIN)

% Choose default command line output for TemperatureGUI
handles.output = hObject;
clear global;
global frame Rotation_Angle cmin cmax
frame = 1;
Rotation_Angle = 0;
cmin = 0;
cmax = 15000;
set(handles.frameno,'String','1')
set(handles.emissivity, 'string', '0.30');
set(handles.lossfactor, 'String', '0.20');
set(handles.exposure, 'String', '1');
set(handles.readout, 'String', '38.9193');
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes TemperatureGUI wait for user response (see UIRESUME)
% uiwait(handles.figure1);
% *** Outputs from this function are returned to the command line.
function varargout = TemperatureGUI_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% *** Executes on button press in show.
function show_Callback(hObject, eventdata, handles)
global A B C D E frame Rotation_Angle Press_Check FilePath FileName
global exp I1 cmin cmax
frame = str2num(char(get(handles.frameno, 'String')));
A = imread([FilePath FileName],frame);
exp = str2num(char(get(handles.exposure, 'String')));
switch exp
    case 1
        B = imread('C1.tif',1); %average background (average background.m)
    case 2
        B = imread('C2.tif',1);
APPENDIX A (continued)

case 0.1
    B = imread('C0.1.tif',1);
case 0.5
    B = imread('C0.5.tif',1);
case 0.125
    B = imread('C0.125.tif',1);
case 0.2
    B = imread('C0.2.tif',1);
otherwise
    h = waitfor(msgbox('Background for the exposure Value not available',
    'Error','error'));
end
C = minus(A,B);
D = imrotate(C,-Rotation_Angle); %rotate by negative of the
value from slider
axes(handles.actual);
% E = imadjust(D); %for adjusting the contrast
imshow(D,jet(cmax));
colorbar;
Press_Check = 1;

% --- Executes on button press in clear.
function clear_Callback(hObject, eventdata, handles)
% hObject    handle to clear (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
    global frame;
    frame = 0;
    axes(handles.actual);
    imshow('hi.png');

% --- Executes on button press in next.
function next_Callback(hObject, eventdata, handles)
% hObject    handle to next (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
    global frame
    frame = str2num(char(get(handles.frameno, 'String')));
    frame = frame +1;
    set(handles.frameno, 'string', frame);
    show_Callback(hObject, eventdata, handles);

% --- Executes on button press in back.
function back_Callback(hObject, eventdata, handles)
% hObject    handle to back (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

global frame
frame = str2num(char(get(handles.frameno, 'String')));  % get the frame number from the textbox and subtract 1 from it,
frame = frame - 1;
set(handles.frameno, 'string', frame);  % write the updated value to the textbox

show_Callback(hObject, eventdata, handles);

% --- Executes on button press in first.
function first_Callback(hObject, eventdata, handles)
% hObject handle to first (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
global frame
frame = 1  % goto first frame
set(handles.frameno, 'string', frame);
show_Callback(hObject, eventdata, handles);

function frameno_Callback(hObject, eventdata, handles)
% hObject handle to frameno (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
show_Callback(hObject, eventdata, handles);

% Hints: get(hObject,'String') returns contents of frameno as text
%        str2double(get(hObject,'String')) returns contents of frameno as a double

function frameno_CreateFcn(hObject, eventdata, handles)
% hObject handle to frameno (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function emissivity_Callback(hObject, eventdata, handles)
% hObject handle to emissivity (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of emissivity as text
%        str2double(get(hObject,'String')) returns contents of emissivity as a double
% --- Executes during object creation, after setting all properties.
function emissivity_CreateFcn(hObject, eventdata, handles)
% hObject    handle to emissivity (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
    get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function lossfactor_Callback(hObject, eventdata, handles)
% hObject    handle to lossfactor (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'String') returns contents of lossfactor as text
%        str2double(get(hObject,'String')) returns contents of lossfactor as a double

% --- Executes during object creation, after setting all properties.
function lossfactor_CreateFcn(hObject, eventdata, handles)
% hObject    handle to lossfactor (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
    get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in crop.
function crop_Callback(hObject, eventdata, handles)
% hObject    handle to crop (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global D I1 rect cmax;
axes(handles.actual);
[I1, rect] =imcrop(D,jet(cmax)); %crop the ROI from the figure
rect = round(rect,0); % rect returns [Xmin Ymin width height]
imshow(I1,jet(cmax));
colorbar;
% figure (4)
% imshow(I1,jet(cmax));
% h = colorbar;
% set(get(h,'title'),'string','Radiation Intensity (counts)', 'Rotation', 90.0);
% figure (5)
% imshow(D,jet(cmax));
% h = colorbar;
% set(get(h,'title'),'string','Radiation Intensity (counts)', 'Rotation', 90.0);

% --- Executes on button press in TemperatureGUI.
function plot_Callback(hObject, eventdata, handles)
% hObject    handle to TemperatureGUI (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global I1 I_LF Temp Tavg Taxis Tsize F_amb inslog LN Em LF exp err;
global Tmax Tmax_pos Tmax_err Distmax frame read_out interframe_time
ConA = 0.384737886295667;
% constants from BB calibration
ConB = 134.490010086082;
ConC = 514979108576438;
Conc2 = 14388;
F_amb = 4.33506105738012E-11;

Em = str2num(char(get(handles.emissivity, 'String')));
% get the values from the constants panel
LF = str2num(char(get(handles.lossfactor, 'String')));
exp = str2num(char(get(handles.exposure, 'String')));
read_out = str2num(char(get(handles.readout, 'String')));
interframe_time = round(exp + read_out,2);
I1 = double(I1);
% global I1 i.e crop data.
I_LF = ((I1 ./ (exp*(1-LF))) - ((1-Em)*F_amb))/Em;
% find all the negative values and make it zero
I_LF(find(I_LF < 0)) = 0;
inslog = ((ConC ./ I_LF + 1));
LN = (ConA.*log(inslog));
T = Conc2 ./ LN - (ConB/ConA) - 273.15;
% find all the negative values and make it zero
T(find(T < 0)) = 0;
Tavg = mean(T);
err = std(T);
axes(handles.processed);
Xaxis = 0:10.24:2800;
% define xaxis from 0 to 2800 at interval of 10.24
Tsize = size(Tavg,2);
% returns number of columns
errorbar(Xaxis(1:Tsize), Tavg, err, 'markersize', 20); % plot the error bar
APPENDIX A (continued)

title('Temperature plot'); %plot
axis([0 inf 600 1400]);
xlabel('Distance from cutting edge (µm)');
ylabel('Temperature (°C)');
datacursormode on;
Tmax = max(Tavg); %find the peak temperature and store it in Tmax
[row column] = find(Tavg == Tmax); %find teh position of the Tmax
Tmax_err = err(row,column);
%corresponding std dev of the Tmax
Distmax = Xaxis(row,column);
%dimension for teh annotation
dim = [.7 .5 .3 .3];
s1 = ('Peak temperature (°C) = ');
s2 = num2str(round(Tmax,0));
s3 = (' ± ');
s4 = num2str(round(Tmax_err,0));
s5 = strcat(s1,s2,s3,s4);
s6 = ('Distance from cutting edge (µm) = ');
s7 = num2str(Distmax);
s8 = strcat(s6,s7);
annotation('textbox',dim,'String',s5,'FitBoxToText','on'); %annotation displays the peak temperature ± std dev and distance from cutting edge
dim2 = [.7 .45 .3 .3];
annotation('textbox',dim2,'String',s8,'FitBoxToText','on');
set(handles.max_temp,'string',round(Tmax,0));
set(handles.max_temp_std,'string',round(Tmax_err,0));
set(handles.dist_cutting,'string',Distmax);

function exposure_Callback(hObject, eventdata, handles)
% hObject    handle to exposure (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of exposure as text
%        str2double(get(hObject,'String')) returns contents of exposure as a double

% --- Executes during object creation, after setting all properties.
function exposure_CreateFcn(hObject, eventdata, handles)
% hObject    handle to exposure (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on slider movement.
function Rot_Angle_Callback(hObject, eventdata, handles)
% hObject    handle to Rot_Angle (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Rotation_Angle
    Rotation_Angle = round(get(handles.Rot_Angle,'Value'),0);
    set(handles.Angle_Rotation,'String',num2str(Rotation_Angle));
    show_Callback(hObject, eventdata, handles);
% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider

% --- Executes during object creation, after setting all properties.
function Rot_Angle_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Rot_Angle (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

% --- Executes on key press with focus on Rot_Angle and none of its controls.
function Rot_Angle_KeyPressFcn(hObject, eventdata, handles)
% hObject    handle to Rot_Angle (see GCBO)
% eventdata  structure with the following fields (see
%   MATLAB.UI.CONTROL.UICONTROL)
%   Key: name of the key that was pressed, in lower case
%   Character: character interpretation of the key(s) that was pressed
%   Modifier: name(s) of the modifier key(s) (i.e., control, shift) pressed
%   handles    structure with handles and user data (see GUIDATA)

% --- Executes on button press in browse.
function browse_Callback(hObject, eventdata, handles)
% hObject    handle to browse (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
APPENDIX A (continued)

global FileName FilePath Force_Data Name_Cell
[FileName,FilePath] = uigetfile('.TIF','Select TIF file');
set(handles.File_Path,'String',FilePath,'horizontalAlignment','right')
set(handles.File_Name,'String',FileName,'horizontalAlignment','left')

% --- Executes on button press in framestart.
function framestart_Callback(hObject, eventdata, handles)
    % hObject    handle to framestart (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    global frame;
    set(handles.frame_start,'String',frame);

function frame_start_Callback(hObject, eventdata, handles)
    % hObject    handle to frame_start (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    % Hints: get(hObject,'String') returns contents of frame_start as text
    %        str2double(get(hObject,'String')) returns contents of frame_start as a double

% --- Executes during object creation, after setting all properties.
function frame_start_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to frame_start (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    empty - handles not created until after all CreateFcns called
    % Hint: edit controls usually have a white background on Windows.
    % See ISPC and COMPUTER.
    if ispc && isequal(get(hObject,'BackgroundColor'),
        get(0,'defaultUicontrolBackgroundColor'))
        set(hObject,'BackgroundColor','white');
    end

% --- Executes on button press in frameend.
function frameend_Callback(hObject, eventdata, handles)
    % hObject    handle to frameend (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    global frame;
    set(handles.frame_end,'String',frame);

function frame_end_Callback(hObject, eventdata, handles)
    % hObject    handle to frame_end (see GCBO)
APPENDIX A (continued)

% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of frame_end as text
% str2double(get(hObject,'String')) returns contents of frame_end as a double

% --- Executes during object creation, after setting all properties.
function frame_end_CreateFcn(hObject, eventdata, handles)
% hObject handle to frame_end (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in plot_all.
function plot_all_Callback(hObject, eventdata, handles)
% hObject handle to plot_all (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
global rect fm_start fm_end index D Vect_Z1
global I1 I_LF Temp Tavg T Xaxis Tsize F_amb inslog LN Em LF exp err T_avgtot T_errtot

peak_val = [255,0,0]/255;
%color for the plot lines in plot3
fm_start = str2num(char(get(handles.frame_start,'String')));
%starting frame number
fm_end = str2num(char(get(handles.frame_end,'String')));
%ending frame number
for index = 1:1:(fm_end-fm_start+1)
%for loop starts from Index 1 to total number of frames selected
    set(handles.frameno,'string',index+fm_start-1);
    show_Callback(hObject, eventdata, handles);
%execute teh function show_Callback as all the processing is done there
    I1 = double(D(rect(2):rect(2)+rect(4)-1,rect(1):rect(1)+rect(3)-1));
%I1 is crop of D, so we take the crop region of interest in the variable rect
defined globally
    ConA = 0.394260851;
    ConB = 125.3933338;
    ConC = 495989937839283;
%calculate the temperature pixel by pixel same as in plot_callback
    Conc2 = 14388;
    F_amb = 9.4369792782478E-12;
    Em = str2num(char(get(handles.emissivity,'String')));
APPENDIX A (continued)

LF = str2num(char(get(handles.lossfactor, 'String')));
exp = str2num(char(get(handles.exposure, 'String')));
read_out = str2num(char(get(handles.readout, 'String')));
interframe_time = round(exp + read_out,2);

% interframe time
I1 = double(I1);
I_LF = ((I1 ./ (exp*(1 - LF))) - ((1 - Em)* F_amb))/Em;
[row column] = find(I_LF < 0);
I_LF(row, column) = 0;
inslog = ((ConC ./ I_LF + 1));
LN = ( ConA .* log(inslog));
T = ConC2 ./ LN - (ConB/ConA) - 273.15;
[row column] = find(T < 0);
T(row, column) = 0;
T_pixelbypixel = [T_pixelbypixel; T];
Tavg = mean(T);
err = std(T);
axes(handles.processed);
Xaxis = 0:10.24:2800;
Tsize = size(Tavg,2);
T_avgtot (index,:) = Tavg;

% the temperature average will be saved in another 2D vector, each row is temp from each frame
T_errtot (index,:) = err;

Corresponding std dev
end
figure('rend', 'painters', 'pos', [0 0 800 600])
% figure size definition
for i = 1 : size(T_avgtot,1)
% plot the frames in 3D
    Vect_Z = ones(1, Tsize)*i;
% definition for z-axis
    plot3(Vect_Z, Xaxis(1:Tsize), T_avgtot(i,:), 'color', pcol)
% plot the temperature using plot3
    title('Framewise Temperature plot', 'fontsize', 20);
% title of the figure, with fontsize
    hold on;
% axis([1 inf 0 800 600 1400]);
% axis minimum and maximum set as default
    ylh = get(gca, 'ylabel');
    gyl = get(ylh);
        % Object Information
    ylp = get(ylh, 'Position');
    set(ylh, 'Rotation', 0, 'Position', ylp, 'VerticalAlignment', 'middle',
        'HorizontalAlignment', 'center')
% alignment of yaxis
    ax = gca;
    ax.YDir = 'reverse';
% reversal of yaxis
    set(get(gca, 'ylabel'), 'rotation', -16);
% rotation of the label of y axis and xaxis
    set(get(gca, 'xlabel'), 'rotation', 15);
    set(gca, 'FontSize', 14)
xticks([1 2 3 4 5 6 7 8 9 10 11 12 13])
  xlabel('Frame number', 'fontsize',18);
  ylabel('Distance from cutting edge (µm)', 'fontsize',18);
  zlabel('Temperature (°C)', 'fontsize',18);
  view(-45,20);
  grid on;
  peak_val(2,fm_start+i-1) = max(T_avgtot(i,:));
end

figure('rend','painters','pos',[0 0 801 601])
for j = 1 : size(T_avgtot,1)
  Vect_Z1 = interframe_time/1000 * ones(1,Tsize)* (j + fm_start);
  %definition for z-axis
  plot3(Vect_Z1, Xaxis(1:Tsize),T_avgtot(j,:), 'color','blue')
  %plot the temperature using plot3
  title('Framewise Temperature plot', 'fontsize',20);
  %title of the figure, with fontsize
  hold on;
  axis([{(fm_start*interframe_time/1000) inf 0 800 600 1400}]);
  %axis minimum and maximum set as default
  ylh = get(gca,'ylabel');
  gyl = get(ylh);
  %Object Information
  ylp = get(ylh, 'Position');
  set(ylh, 'Rotation',0, 'Position',ylp, 'VerticalAlignment','middle',
     'HorizontalAlignment','center') %alignment of yaxis
  ax = gca;
  ax.YDir = 'reverse';
  %reversal of yaxis
  set(get(gca,'ylabel'), 'rotation',-16);
  %rotation of the label of y axis and xaxis
  set(get(gca,'xlabel'), 'rotation',15);
  set(gca, 'FontSize', 14)
  xticks('auto');
  % xticks([0:round(interframe_time/1000,2):round((fm_start
+13)*interframe_time/1000,2)]);
  % xticks([0 fm_start*interframe_time/1000*1
  (fm_start+1)*interframe_time/1000 (fm_start+2)*interframe_time/1000
  fm_start*interframe_time/1000*4 fm_start*interframe_time/1000*5
  fm_start*interframe_time/1000*6 fm_start*interframe_time/1000*7
  fm_start*interframe_time/1000*8 fm_start*interframe_time/1000*9
  fm_start*interframe_time/1000*10 fm_start*interframe_time/1000*11
  fm_start*interframe_time/1000*12 fm_start*interframe_time/1000*13]);
  xlabel('Time (s)', 'fontsize',18);
  ylabel('Distance from cutting edge (µm)', 'fontsize',18);
  zlabel('Temperature (°C)', 'fontsize',18);
  view(-45,20);
  grid on;
end
% --- Executes on button press in plot_avg.
%function plot_avg_Callback(hObject, eventdata, handles)
% hObject    handle to plot_avg (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global T_avgtot_average T_errtot_average T_vartot T_vartot_average

% Define new set of global variables to find the average of all the frames

Tmaxavg = max(T_avgtot_average);

Tavgsize = size(T_avgtot_average,2);

dim = [.45 .5 .3 .3];
s1 = {'Peak temperature (°C) = '};
s2 = num2str(round(Tmaxavg,0));
s3 = {' ± '};
s4 = num2str(round(Tmaxavg_err,0));
s5 = strcat(s1,s2,s3,s4);
s6 = {'Distance from cutting edge (µm) = '};
s7 = num2str(Distmaxavg);
s8 = strcat(s6,s7);
annotation('textbox',dim,'String',s5,'FitBoxToText','on');
dim2 = [.45 .41 .3 .3];
APPENDIX A (continued)

annotation('textbox',dim2,'String',s8,'FitBoxToText','on');

function readout_Callback(hObject, eventdata, handles)
% hObject    handle to readout (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of readout as text
%        str2double(get(hObject,'String')) returns contents of readout as a double

% --- Executes during object creation, after setting all properties.
function readout_CreateFcn(hObject, eventdata, handles)
% hObject    handle to readout (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
                  get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in saveall.
function saveall_Callback(hObject, eventdata, handles)
% hObject    handle to saveall (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global FileName FilePath dir
dir = [FilePath FileName(1:(length(FileName)-4))]
YorN = exist(dir);
if YorN == 7
    mkdir(dir);
end

global rect fm_start fm_end index D
global I1 I_LF Temp Tavg Taxis Tsize F_amb inslog LN Em LF exp err T_avgtot
T_errtot
global Tmax Tmax_pos Tmax_err Distmax T_pixelbypixel read_out interframe_time
pcol = [255,0,0]/255;
%color for the plot lines in plot3
fm_start = str2num(char(get(handles.frame_start, 'String')));
%starting frame number
fm_end = str2num(char(get(handles.frame_end, 'String')));
%ending frame number
for index = 1:1:(fm_end-fm_start+1)
%for loop starts from Index 1 to total number of frames selected
    set(handles.frameno, 'string', index+fm_start-1);
end
show_Callback(hObject, eventdata, handles);
%execute teh function show_Callback as all the processing is done there
I1 = double(D(rect(2):rect(2)+rect(4)-1,rect(1):rect(1)+rect(3)-1));
%I1 is crop of D, so we take the crop region of interest in the variable rect
defined globaly
ConA = 0.394260851;
ConB = 125.3933338;
ConC = 495989937839283;
%calculate the temperature pixel by pixel same as in plot_callback
Conc2 = 14388;
F_amb = 9.4369792782478E-12;
Em = str2num(char(get(handles.emissivity, 'String')));
LF = str2num(char(get(handles.lossfactor, 'String')));
exp = str2num(char(get(handles.exposure, 'String')));
read_out = str2num(char(get(handles.readout, 'String')));
interframe_time = round(exp + read_out,2);
%interframe time
I1 = double(I1);
I_LF = ((I1 ./(exp*(1 - LF))) - (( 1 - Em)* F_amb))/Em;
[row column] = find(I_LF < 0);
I_LF(row,column) = 0;
inslog = ((ConC ./ I_LF + 1));
LN = ( ConA .* log(inslog));
T = Conc2 ./ LN - (ConB/ConA) - 273.15;
[row column] = find(T < 0);
T(row,column) = 0;
T_pixelbypixel = [T_pixelbypixel; T;];
Tavg = mean(T);
err = std(T);
axes(handles.processed);
Xaxis = 0:10.24:2800;
Tsize = size(Tavg,2);
T_avgtot (index,:) = Tavg;
%the temperature average will be saved in another 2D vector, each row is temp
from each frame
T_errtot (index,:) = err;
%Corresponding std dev
end

figure('rend','painters','pos',[0 0 800 600])
%figure size definition
for i = 1 : size(T_avgtot,1)
%plot the frames in 3D
   Vect_Z = ones(1,Tsize)*i;
%definition for z-axis
   plot3(Vect_Z,Xaxis(1:Tsize),T_avgtot(i,:),'color',pcol)
%plot the temperature using plot3
%   title('Framewise Temperature plot','fontsize',20);
%title of the figure, with fontsize
   hold on;
   axis([1 inf 0 800 600 1400]);
%axis minimum and maximum set as default
APPENDIX A (continued)

ylh = get(gca,'ylabel');
gyl = get(ylh);

% Object Information

    ylp = get(ylh,'Position');
    set(ylh,'Rotation',0,'Position',ylp,'VerticalAlignment','middle','HorizontalAlignment','center')  %alignment of yaxis

    ax = gca;
    ax.YDir = 'reverse';

%reversal of yaxis

    set(get(gca,'ylabel'),'rotation',-16);
%rotation of the label of y axis and xaxis

    set(get(gca,'xlabel'),'rotation',15);

    set(gca,'FontSize', 14)
    xticks('auto');

end
dirfile_name = [dir '\' FileName(1:(length(FileName)-4)) '_framenumber.jpg'];
saveas(gcf,dirfile_name);
close;

figure('rend','painters','pos',[0 0 801 601])

for j = 1 : size(T_avgtot,1)

    Vect_Z1 = interframe_time/1000 * ones(1,Tsize)*(j-2 + fm_start);
%definition for z-axis

    plot3(Vect_Z1, Xaxis(1:Tsize),T_avgtot(j,:),color,'blue')
%plot the temperature using plot3

%   title('Framewise Temperature plot','fontsize',20);
%title of the figure, with fontsize

    hold on;

    axis([((fm_start -2) *interframe_time/1000) inf 0 800 600 1400]);
%axis minimum and maximum set as default

    ylh = get(gca,'ylabel');
gyl = get(ylh);

% Object Information

    ylp = get(ylh,'Position');
    set(ylh,'Rotation',0,'Position',ylp,'VerticalAlignment','middle','HorizontalAlignment','center')  %alignment of yaxis

    ax = gca;
    ax.YDir = 'reverse';

%reversal of yaxis

    set(get(gca,'ylabel'),'rotation',-16);
%rotation of the label of y axis and xaxis

    set(get(gca,'xlabel'),'rotation',15);

    set(gca,'FontSize', 14)
    xticks('auto');
% xticks([0 round(interframe_time*1,0) round(interframe_time*2,0)
    round(interframe_time*3,0) round(interframe_time*4,0)
    round(interframe_time*5,0))
xlabel('Time (s)', 'fontsize',18);
ylabel('Distance from cutting edge (µm)', 'fontsize',18);
zlabel('Temperature (°C)', 'fontsize',18);
view(-45,20);
grid on;
end
dirfile_name = [dir '\ FileName(1:(length(FileName)-4)) '_frametime.jpg'];
saveas(gcf,dirfile_name);
close;

global T_avgtot_average T_errtot_average T_vartot T_vartot_average
%Define new set of global variables to find the average of all the frames
global Tmaxavg Tmaxavg_err Distmaxavg Tavgsize
%standard deviation of the average is taken by finding the sqrt of average of
variance.
figure('rend','painters','pos',[0 0 600 400])
T_avgtot_average = mean(T_avgtot);
%average of frame average temperature
T_vartot = T_errtot .^2;
%finding the varience from the std dev 2D vector
T_vartot_average = mean(T_vartot);
%finding average of teh varience columnwise
T_errtot_average = T_vartot_average.^0.5;
%taking sqrt of the varience, i.e the new std dev of average of average
T_errtot_average = std(T_pixelbypixel);
%finding the standard deviation of the varience
T_errtot_average = T_errtot_average ./ (size(T,1)^0.5);
%find the peak temperature from the average plot
Tavgsize = size(T_avgtot_average,2);
% determine the size of the Taverage, to correspond it with the x axis label
[row1 column1] = find(T_avgtot_average == Tmaxavg);
%position of the peak temperature, to find the corresponding std dev
Tmaxavg_err = T_errtot_average(row1,column1);
Distmaxavg = Xaxis(row1,column1);
%find the corresponding distance of teh peak temperature from the cutting
edge
errorbar(Xaxis(1:Tavgsize), T_avgtot_average, T_errtot_average, 'markersize',
    20);
% title('Average Temperature plot');
axis([0 800 600 14000]);
xlabel('Distance from cutting edge (µm)', 'FontSize',16);
ylabel('Temperature (°C)', 'FontSize',16);
x = get(gca, 'XTick');
set(gca, 'FontSize', 16);
datacursormode on;
dim = [.45 .5 .3 .3];
s1 = {'Peak temperature (°C) = '};
APPENDIX A (continued)

s2 = num2str(round(Tmaxavg,0));
s3 = (' ± ');
s4 = num2str(round(Tmaxavg_err,0));
s5 = strcat(s1,s2,s3,s4);
s6 = ('Distance from cutting edge (µm) = ');
s7 = num2str(Distmaxavg);
s8 = strcat(s6,s7);
annotation('textbox',dim,'String',s5,'FitBoxToText','on');
dim2 = [.45 .41 .3 .3];
annotation('textbox',dim2,'String',s8,'FitBoxToText','on');

dirfile_name = [dir ' FileName(1:(length(FileName)-4)) '_avgplot.jpg'];
saveas(gcf,dirfile_name);
close;
dirfile_name = [dir ' FileName(1:(length(FileName)-4)) '_average temp.xlsx'];
xlswrite(dirfile_name,Xaxis(1:Tavgsize), T_avgtot_average, T_errtot_average);

% --- Executes on slider movement.
function coloradjust_Callback(hObject, eventdata, handles)
    % hObject    handle to coloradjust (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    global cmax
cmax = round(get(handles.coloradjust,'Value'),0);
    show_Callback(hObject, eventdata, handles);
    % Hints: get(hObject,'Value') returns position of slider
    % get(hObject,'Min') and get(hObject,'Max') to determine range of slider

% --- Executes during object creation, after setting all properties.
function coloradjust_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to coloradjust (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    empty - handles not created until after all CreateFcns called

    % Hint: slider controls usually have a light gray background.
    if isequal(get(hObject,'BackgroundColor'),
        get(0,'defaultUicontrolBackgroundColor'))
        set(hObject,'BackgroundColor',[.9 .9 .9]);
    end

% --- Executes on button press in closefig.
function closefig_Callback(hObject, eventdata, handles)
    % hObject    handle to closefig (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    close all;
% --- Executes on button press in trig.
function trig_Callback(hObject, eventdata, handles)
    global peak_val interframe_time dir FileName
    for i = 1:50
        peak_val(1,i) = interframe_time/1000 * (i-1);
    end
dirfile_name = [dir '\' FileName(1:(length(FileName)-4)) '_peaktemp.xlsx']
    xlswrite(dirfile_name,peak_val);
APPENDIX B

FORCE DATA PROCESSING MATLAB CODE

function varargout = Force(varargin)
% FORCE MATLAB code for Force.fig
% FORCE, by itself, creates a new FORCE or raises the existing
% singleton*. %
% H = FORCE returns the handle to a new FORCE or the handle to
% the existing singleton*. %
% FORCE('CALLBACK', hObject, eventdata, handles, ...) calls the local
% function named CALLBACK in FORCE.M with the given input arguments.
% FORCE('Property','Value',...) creates a new FORCE or raises the
% existing singleton*. Starting from the left, property value pairs are
% applied to the GUI before Force_OpeningFcn gets called. An
% unrecognized property name or invalid value makes property application
% stop. All inputs are passed to Force_OpeningFcn via varargin.
% *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
% instance to run (singleton)".
% See also: GUIDE, GUIDATA, GUIDATA
% Edit the above text to modify the response to help Force
% Last Modified by GUIDE v2.5 01-Nov-2017 15:40:04

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @Force_OpeningFcn, ...
    'gui_OutputFcn', @Force_OutputFcn, ...
    'gui_LayoutFcn', [], ..., ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Force is made visible.
function Force_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OpeningFcn
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
APPENDIX B (continued)

% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Force (see VARARGIN)
global X_crop Y_crop Z_crop channel_crop yslide Ylimshow  % Set the
default values
X_crop = 0;
Y_crop = 0;
Z_crop = 0;
yslide = 600;
channel_crop = 0;
set(handles.sens_cutting, 'String', -100);
set(handles.sens_thrust, 'String', 100);
set(handles.sens_X, 'String', 100);
set(handles.sens_channel, 'String', 1);
set(handles.freq, 'String', 10000);
set(handles.y_check, 'value', 1);
set(handles.z_check, 'value', 1);
set(handles.x_check, 'value', 1);
set(handles.channel_check, 'value', 1)
set(handles.rakeangle, 'String', -5);
Ylimshow = [-100:1:((4*yslide/5)-100)];
set(handles.y_posi, 'string', 0);

% Choose default command line output for Force
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Force wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = Force_OutputFcn(hObject, eventdata, handles)
% varargout   cell array for returning output args (see VARARGOUT);
% hObject     handle to figure
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in Show.
function Show_Callback(hObject, eventdata, handles)
% hObject     handle to Show (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global X_force Y_force Z_force Time_s Fs T L FilePath FileName
global check_cutting check_thrust check_X check_channel cut_sens thr_sens
x_sens channel_sens channel_force
check_cutting = get(handles.y_check, 'Value');
check_thrust = get(handles.z_check, 'Value');
check_X = get(handles.x_check, 'Value');
check_channel = get(handles.channel_check, 'Value');

% checkbox values 0 for 'no' 1 for 'yes'

% set(handles.y_check,'Enable','off');
% set(handles.z_check,'Enable','off');
% set(handles.x_check,'Enable','off');
% set(handles.channel_check,'Enable','off');

cut_sens = str2num(char(get(handles.sens_cutting,'String')));
thr_sens = str2num(char(get(handles.sens_thrust, 'String')));
x_sens = str2num(char(get(handles.sens_X, 'String')));
channel_sens = str2num(char(get(handles.sens_channel, 'String')));
X_force = (x_sens .* xlsread([FilePath FileName], 'B:B'));
Y_force = (cut_sens .* xlsread([FilePath FileName], 'D:D'));
Z_force = (thr_sens .* xlsread([FilePath FileName], 'F:F'));
channel_force = (channel_sens .* xlsread([FilePath FileName], 'H:H'));

Fs = str2num(char(get(handles.freq, 'String')));  % frequency of the signal
T = 1/Fs;                                           % sample period
L = size(Y_force,1);                                % total no of samples
Time_s = ((0:L-1)*T)';
axes(handles.Plot_graph);
hold on;
cla;
plot(Time_s,X_force); % Plot the original
plot(Time_s,Y_force);
plot(Time_s,Z_force);
if check_channel == 1
plot(Time_s,channel_force);
end
hold off;
waitfor(msgbox('Zero correct the signal before cropping', 'Warning','warn'));

% --- Executes on button press in Crop.
function Crop_Callback(hObject, eventdata, handles)
% hObject    handle to Crop (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global Time_s Fs x y Time_min Time_max row column X_force Y_force Z_force
global X_crop Y_crop Z_crop channel_crop Time_crop

% waitfor(msgbox('Crop the region for having a closer look', 'What to do?', 'help'));
% check_cutting = get(handles.y_check, 'Value');
% check_thrust = get(handles.z_check, 'Value');
% check_X = get(handles.x_check, 'Value');
% check_channel = get(handles.channel_check, 'Value');
axes(handles.Plot_graph);
[x,y] = ginput(2);
[row column] = find(Time_s == x(1));
Time_min = Time_s(row,column);
[row column] = find(Time_s == x(2));
Time_max = Time_s(row,column);
X_crop = X_force(int32(Fs*x(1)) : int32(Fs*x(2)));
Y_crop = Y_force(int32(Fs*x(1)) : int32(Fs*x(2)));
Z_crop = Z_force(int32(Fs*x(1)) : int32(Fs*x(2)));
if check_channel == 1
channel_crop = channel_force(int32(Fs*x(1)) : int32(Fs*x(2)));
end
Time_crop = Time_s(1:size(X_crop,1));
axis([x(1) x(2) -inf inf]);
cla;
hold on;
plot(Time_s,X_force);
plot(Time_s,Y_force);
plot(Time_s,Z_force);
if check_channel = 1
plot(Time_s,channel_force);
end
hold off;
% Time_min = Time_s(x(1));
% Time_max = Time_s(x(2));

function Trig_Callback(hObject, eventdata, handles)
% hObject    handle to Trig (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Fs T L Time_s channel_force i Y_force start_trig end_trig channel_crop
X_force X_crop Y_crop Z_trig Z_crop Time_crop
global peak_val Z_force

[FileName1,FilePath1] = uigetfile('.xlsx','Select excel file for peak temp');
peak_val = xlsread([FilePath1 FileName1]);

Fs = str2num(char(get(handles.freq,'String')));
T = 1/Fs;  % frequency of the signal
L = size(Y_force,1);  % total no of samples
Time_s = ((0:L-1)*T)';
axes(handles.Plot_graph);
i = find(channel_force >= 2);
start_trig = i(1,1);
end_trig = i(size(i,1));
X_crop = X_force(start_trig : end_trig);
Y_crop = Y_force(start_trig : end_trig);
Z_crop = Z_force(start_trig : end_trig);
channel_crop = channel_force(start_trig : end_trig);
Time_crop = ((0:(size(i,1)-1))*T)';
axes(handles.Plot_graph);
cla reset;
hold on;
% --- Executes on button press in trig_crop.
function trig_crop_Callback(hObject, eventdata, handles)
% hObject handle to trig_crop (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

global xt yt X_crop Y_crop Z_crop Time_crop Fs peak_val p
axes(handles.Plot_graph);
[xt,yt] = ginput(2);
X_crop = X_crop(int32(Fs*xt(1)) : int32(Fs*xt(2)));
Y_crop = Y_crop(int32(Fs*xt(1)) : int32(Fs*xt(2)));
Z_crop = Z_crop(int32(Fs*xt(1)) : int32(Fs*xt(2)));
Time_crop = Time_crop(int32(Fs*xt(1)) : int32(Fs*xt(2)));
figure(1);
cla reset;
hold on;
xlim([xt(1) xt(2)]);
yyaxis left
ylabel('Force (N)');
ylim([0 600]);
plot(Time_crop, Y_crop,'r');
plot(Time_crop, Z_crop,'g');
plot(Time_crop, X_crop,'c');

yyaxis right
ylim([500 1400])
ylabel('Temperature (°C)')
plot((peak_val(1,:)), peak_val(2,:), 'b*');
legend('Cutting Force (N)','Thrust Force (N)','X force (N)','Peak Temperature');
hold off;

axes(handles.trig_plot);
cla reset;
hold on;
xlim([xt(1) xt(2)]);
yyaxis left
ylabel('Force (N)');
ylim([0 600]);
plot(Time_crop, Y_crop,'r');
plot(Time_crop, Z_crop,'g');
plot(Time_crop, X_crop,'c');

yyaxis right
ylim([500 1400])
ylabel('Temperature (°C)')
plot((peak_val(1,:)), peak_val(2,:), 'b*');
legend('Cutting Force (N)', 'Thrust Force (N)', 'X force (N)', 'Peak Temperature');
hold off;

% p = polyfit(Y_crop,Time_crop,8)

%% --- Executes on button press in Zerocorrect.
function Zerocorrect_Callback(hObject, eventdata, handles)
% hObject    handle to Zerocorrect (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global Time_s Fs x1 y1 X_force Y_force Z_force X_cor_avg Y_cor_avg Z_cor_avg
channel_crop channel_cor_avg

peak_val
% check_cutting = get(handles.y_check, 'Value');
% check_thrust = get(handles.z_check, 'Value');
% check_X = get(handles.x_check, 'Value');

% waitfor(msgbox('Select limits for zero correction', 'What to do?','help'));
check_channel = get(handles.channel_check, 'Value');
axes(handles.Plot_graph);
[x1,y1] = ginput(2);
if check_X == 1
    X_cor_avg = mean(X_force(int32(Fs*x1(1)) : int32(Fs*x1(2))));
    X_force = X_force - X_cor_avg;
end
if check_cutting == 1
    Y_cor_avg = mean(Y_force(int32(Fs*x1(1)) : int32(Fs*x1(2))));
    Y_force = Y_force - Y_cor_avg;
end
if check_thrust == 1
    Z_cor_avg = mean(Z_force(int32(Fs*x1(1)) : int32(Fs*x1(2))));
    Z_force = Z_force - Z_cor_avg;
end
if check_channel == 1
    channel_cor_avg = mean(channel_force(int32(Fs*x1(1)) : int32(Fs*x1(2))));
    channel_force = channel_force - channel_cor_avg;
end
axes(handles.Plot_graph);
cla;
hold on;
plot(Time_s,X_force);
plot(Time_s,Y_force);
plot(Time_s,Z_force);
if check_channel == 1
    plot(Time_s,channel_force);
end
hold off;
% set(handles.y_check,'Enable','off');
% set(handles.z_check,'Enable','off');
% set(handles.x_check,'Enable','off');
% set(handles.channel_check,'Enable','off');

% --- Executes on button press in selectpoints.
function selectpoints_Callback(hObject, eventdata, handles)
% hObject    handle to selectpoints (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Time_s T Fs x2 y2 Time_min Time_max row column X_force Y_force Z_force
X_crop Y_crop Z_crop Time_crop
global channel_crop channel_force yslide peak_val
global check_cutting check_thrust check_X check_channel xtox
% check_cutting = get(handles.y_check, 'Value');
% check_thrust = get(handles.z_check, 'Value');
% check_X = get(handles.x_check, 'Value');
% waitfor(msgbox('Select starting and ending', 'What to do?','help'));
check_channel = get(handles.channel_check, 'Value');
axes(handles.Plot_graph);
[x2,y2] = ginput(2);
xtox = round((Fs*x2(2) - Fs*x2(1))/3,0);
X_crop = X_crop(int32(Fs*x2(1)-xtox) : int32(Fs*x2(2))+xtox);
Y_crop = Y_crop(int32(Fs*x2(1)-xtox) : int32(Fs*x2(2))+xtox);
Z_crop = Z_crop(int32(Fs*x2(1)-xtox) : int32(Fs*x2(2))+xtox);
if check_channel == 1
    channel_crop = channel_crop(int32(Fs*x2(1)-xtox) : int32(Fs*x2(2))+xtox);
end
Time_crop = Time_crop(int32(Fs*x2(1)-xtox) : int32(Fs*x2(2))+xtox);
axes(handles.Plot_graph);
% axis([-inf inf -yslide yslide]);
ax = gca;
cla;
hold on;
plot(Time_crop,X_crop);
plot(Time_crop,Y_crop);
plot(Time_crop,Z_crop);
if check_channel == 1
    plot(Time_crop,channel_crop);
end
yyaxis right
plot((peak_val(1,:)), peak_val(2,:), 'g*');
hold off;

% --- Executes on button press in average.
function average_Callback(hObject, eventdata, handles)
% hObject    handle to average (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Fs x3 y3 X_avg Y_avg Z_avg X_crop Y_crop Z_crop X_std Y_std Z_std xtox
global Force_normal Force_friction mu mu_avg mu_std channel_crop channel_avg channel_std
global check_cutting check_thrust check_X check_channel rake_angle
% waitfor(msgbox('Select region for averaging', 'What to do?','help'));

% check_cutting = get(handles.y_check, 'Value');
% check_thrust = get(handles.z_check, 'Value');
% check_X = get(handles.x_check, 'Value');
rake_angle = str2num(char(get(handles.rakeangle, 'String')));
check_channel = get(handles.channel_check, 'Value');
axes(handles.Plot_graph);
[x3,y3] = ginput(2);
X_avg = mean(X_crop(int32(Fs*x3(1)+xtox) : int32(Fs*x3(2)+xtox)));
Y_avg = mean(Y_crop(int32(Fs*x3(1)+xtox) : int32(Fs*x3(2)+xtox)));
Z_avg = mean(Z_crop(int32(Fs*x3(1)+xtox) : int32(Fs*x3(2)+xtox)));

X_std = std(X_crop(int32(Fs*x3(1)+xtox) : int32(Fs*x3(2)+xtox)));
Y_std = std(Y_crop(int32(Fs*x3(1)+xtox) : int32(Fs*x3(2)+xtox)));
Z_std = std(Z_crop(int32(Fs*x3(1)+xtox) : int32(Fs*x3(2)+xtox)));

Force_normal = (cosd(rake_angle) .* Y_crop) - (sind(rake_angle) .* Z_crop);
Force_friction = (sind(rake_angle) .* Y_crop) + (cosd(rake_angle) .* Z_crop);
mu = Force_friction ./ Force_normal;
mu_avg = mean(mu(int32(10000*x3(1)+xtox) : int32(10000*x3(2)+xtox)));
mu_std = std(mu(int32(10000*x3(1)+xtox) : int32(10000*x3(2)+xtox)));
set(handles.avg_cuttingforce, 'String', round(Y_avg,0));
set(handles.avg_thrustforce, 'String', round(Z_avg,0));
set(handles.avg_xforce, 'String', round(X_avg,0));
set(handles.avg_friction, 'String', round(mu_avg,2));

set(handles.std_cuttingforce, 'String', round(Y_std,0));
set(handles.std_thrustforce, 'String', round(Z_std,0));
set(handles.std_friction, 'String', round(mu_std,2));
set(handles.std_xforce, 'String', round(X_std,2));

if check_channel == 1
    channel_avg = mean(channel_crop(int32(Fs*x3(1)+xtox) : int32(Fs*x3(2)+xtox)));
    channel_std = std(channel_crop(int32(Fs*x3(1)+xtox) : int32(Fs*x3(2)+xtox)));
    set(handles.avg_chan, 'String', round(channel_avg,0));
    set(handles.std_channel, 'String', round(channel_std,0));
end

% Save_plots_Callback(hObject, eventdata, handles);

% --- Executes on button press in plotsep.
function plotsep_Callback(hObject, eventdata, handles)
% hObject    handle to plotsep (see GCBO)
APPENDIX B (continued)

% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)
global Time_crop Y_crop Z_crop mu  channel_crop yslide xtox
global check_cutting check_thrust check_X check_channel
% check_cutting = get(handles.y_check, 'Value');
% check_thrust = get(handles.z_check, 'Value');
% check_X = get(handles.x_check, 'Value');

axes(handles.cutting);
axis([-inf inf -yslide yslide]);
ax1 = gca;
ax1.YAxisLocation = 'origin';
ax1.XAxisLocation = 'origin';
title('Cutting force');
xlabel('Time (s)');
ylabel('Cutting Force (N)');
set(gca, 'YTick', [-yslide : yslide/6 : yslide]);
cla;
hold on;
plot(Time_crop,Y_crop);
hold off;

axes(handles.thrust);
axis([-inf inf -yslide yslide]);
ax2 = gca;
ax2.YAxisLocation = 'origin';
ax2.XAxisLocation = 'origin';
title('Thrust force');
xlabel('Time (s)');
ylabel('Thrust Force (N)');
set(gca, 'YTick', [-yslide : yslide/6 : yslide]);
cla;
hold on;
plot(Time_crop,Z_crop, 'red');
hold off;

axes(handles.friction);
axis([[Time_crop(xtox) Time_crop(size(Time_crop,1)-xtox) -5 5]]);
ax3 = gca;
ax3.YAxisLocation = 'origin';
ax3.XAxisLocation = 'origin';
title('Coefficient of Friction');
xlabel('Time (s)');
ylabel('Coefficient of Friction');
set(gca, 'YTick', [-5 : 1 : 5]);
cla;
hold on;
plot(Time_crop,mu, 'green');
hold off;

check_channel = get(handles.channel_check, 'Value');
if check_channel == 1
axes(handles.channel);

96
APPENDIX B (continued)

axis([-inf inf -yslide yslide]);
ax2 = gca;
ax2.YAxisLocation = 'origin';
ax2.XAxisLocation = 'origin';
title('Channel X');
xlabel('Time (s)');
ylabel('Channel X');
set(gca,'YTick',[-yslide : yslide/6 : yslide]);
cla;
hold on;
plot(Time_crop,channel_crop,'magenta');
hold off;
end

if check_X == 1
    axes(handles.X_fft);
    axis([-inf inf -yslide yslide]);
    ax2 = gca;
    ax2.YAxisLocation = 'origin';
    ax2.XAxisLocation = 'origin';
title('X force');
xlabel('Time (s)');
ylabel('X Force (N)');
set(gca,'YTick',[-yslide : yslide/6 : yslide]);
cla;
hold on;
plot(Time_crop,Z_crop,'yellow');
hold off;
end

function avg_cuttingforce_Callback(hObject, eventdata, handles)
    % hObject    handle to avg_cuttingforce (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    % Hints: get(hObject,'String') returns contents of avg_cuttingforce as text
    %        str2double(get(hObject,'String')) returns contents of avg_cuttingforce as a double

    % --- Executes during object creation, after setting all properties.
    function avg_cuttingforce_CreateFcn(hObject, eventdata, handles)
        % hObject    handle to avg_cuttingforce (see GCBO)
        % eventdata  reserved - to be defined in a future version of MATLAB
        % handles    empty - handles not created until after all CreateFcns called
        % Hint: edit controls usually have a white background on Windows.
        % See ISPC and COMPUTER.
        if ispc && isequal(get(hObject,'BackgroundColor'),
            get(0,'defaultUicontrolBackgroundColor'))
            set(hObject,'BackgroundColor','white');
        end

97
function avg_thrustforce_Callback(hObject, eventdata, handles)
% hObject    handle to avg_thrustforce (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of avg_thrustforce as text
%        str2double(get(hObject,'String')) returns contents of avg_thrustforce as a double

% --- Executes during object creation, after setting all properties.
function avg_thrustforce_CreateFcn(hObject, eventdata, handles)
% hObject    handle to avg_thrustforce (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function avg_friction_Callback(hObject, eventdata, handles)
% hObject    handle to avg_friction (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of avg_friction as text
%        str2double(get(hObject,'String')) returns contents of avg_friction as a double

% --- Executes during object creation, after setting all properties.
function avg_friction_CreateFcn(hObject, eventdata, handles)
% hObject    handle to avg_friction (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function std_cuttingforce_Callback(hObject, eventdata, handles)
% hObject    handle to std_cuttingforce (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
APPENDIX B (continued)

% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of std_cuttingforce as text
% str2double(get(hObject,'String')) returns contents of std_cuttingforce as a double

% --- Executes during object creation, after setting all properties.
function std_cuttingforce_CreateFcn(hObject, eventdata, handles)
    hObject    handle to std_cuttingforce (see GCBO)
    eventdata  reserved - to be defined in a future version of MATLAB
    handles    structure with handles and user data (see GUIDATA)

    % Hint: edit controls usually have a white background on Windows.
    %       See ISPC and COMPUTER.
    if ispc && isequal(get(hObject,'BackgroundColor'),
        get(0,'defaultUicontrolBackgroundColor'))
        set(hObject,'BackgroundColor','white');
    end

function std_thrustforce_Callback(hObject, eventdata, handles)
    hObject    handle to std_thrustforce (see GCBO)
    eventdata  reserved - to be defined in a future version of MATLAB
    handles    structure with handles and user data (see GUIDATA)

    % Hints: get(hObject,'String') returns contents of std_thrustforce as text
    % str2double(get(hObject,'String')) returns contents of std_thrustforce as a double

% --- Executes during object creation, after setting all properties.
function std_thrustforce_CreateFcn(hObject, eventdata, handles)
    hObject    handle to std_thrustforce (see GCBO)
    eventdata  reserved - to be defined in a future version of MATLAB
    handles    structure with handles and user data (see GUIDATA)

    % Hint: edit controls usually have a white background on Windows.
    %       See ISPC and COMPUTER.
    if ispc && isequal(get(hObject,'BackgroundColor'),
        get(0,'defaultUicontrolBackgroundColor'))
        set(hObject,'BackgroundColor','white');
    end

function std_friction_Callback(hObject, eventdata, handles)
    hObject    handle to std_friction (see GCBO)
    eventdata  reserved - to be defined in a future version of MATLAB
    handles    structure with handles and user data (see GUIDATA)

    % Hints: get(hObject,'String') returns contents of std_friction as text
APPENDIX B (continued)

% str2double(get(hObject,'String')) returns contents of std_friction
% as a double

% --- Executes during object creation, after setting all properties.
function std_friction_CreateFcn(hObject, eventdata, handles)
% hObject    handle to std_friction (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

% --- Executes on button press in fftbutton.
function fftbutton_Callback(hObject, eventdata, handles)
% hObject    handle to fftbutton (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

global Fs Time_s L t X_force Z_force Time_crop Z P1 P2 f X Y Y_force channel channel_force

% check_cutting = get(handles.y_check, 'Value');
% check_thrust = get(handles.z_check, 'Value');
% check_X = get(handles.x_check, 'Value');
check_channel = get(handles.channel_check, 'Value');

Z = fft(Z_force);
L = size(Z_force,1);
P2 = abs(Z/L);
P1 = P2(1:L/2+1);
P1(2:end-1) = 2*P1(2:end-1);
f = (Fs*(0:(L/2))/L)';
axes(handles.thrust_fft);
datacursormode on;
plot(f,P1);

Y = fft(Y_force);
L = size(Y_force,1);
P2 = abs(Y/L);
P1 = P2(1:L/2+1);
P1(2:end-1) = 2*P1(2:end-1);
f = (Fs*(0:(L/2))/L)';
axes(handles.cutting_fft);
datacursormode on;
plot(f,P1);
APPENDIX B (continued)

```matlab
%axis ([0 1000 0 1000]);
title('Single-Sided Amplitude Spectrum of Cutting force','fontsize', 8);
xlabel('f (Hz)');
ylabel('Amplitude');
X = fft(X_force);
L = size(X_force,1);
P2 = abs(X/L);
P1 = P2(1:L/2+1);
P1(2:end-1) = 2*P1(2:end-1);
f = (Fs*(0:(L/2))/L)';
axes(handles.channel_fft);
datacursormode on;
plot(f,P1);

%axis ([0 1000 0 1000]);
title('Single-Sided Amplitude Spectrum of X force','fontsize', 8);
xlabel('f (Hz)');
ylabel('Amplitude');
if check_channel == 1
channel = fft(channel_force);
L = size(channel_force,1);
P2 = abs(X/L);
P1 = P2(1:L/2+1);
P1(2:end-1) = 2*P1(2:end-1);
f = (Fs*(0:(L/2))/L)';
axes(handles.channel_fft);
datacursormode on;
plot(f,P1);
%axis ([0 1000 0 1000]);
title('Single-Sided Amplitude Spectrum of Channel X','fontsize', 8);
xlabel('f (Hz)');
ylabel('Amplitude');
end

% --- Executes on button press in y_check.
function y_check_Callback(hObject, eventdata, handles)
% hObject    handle to y_check (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of y_check

% --- Executes on button press in z_check.
function z_check_Callback(hObject, eventdata, handles)
% hObject    handle to z_check (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hint: get(hObject,'Value') returns toggle state of z_check
```
APPENDIX B (continued)

% --- Executes on button press in x_check.
function x_check_Callback(hObject, eventdata, handles)
% hObject    handle to x_check (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles   structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of x_check

% --- Executes on button press in channel_check.
function channel_check_Callback(hObject, eventdata, handles)
% hObject    handle to channel_check (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of channel_check

function avg_chan_Callback(hObject, eventdata, handles)
% hObject    handle to avg_channel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of avg_channel as text
%        str2double(get(hObject,'String')) returns contents of avg_channel as a double

% --- Executes during object creation, after setting all properties.
function avg_channel_CreateFcn(hObject, eventdata, handles)
% hObject    handle to avg_channel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function std_channel_Callback(hObject, eventdata, handles)
% hObject    handle to std_channel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of std_channel as text
%        str2double(get(hObject,'String')) returns contents of std_channel as a double
APPENDIX B (continued)

% --- Executes during object creation, after setting all properties.
function std_channel_CreateFcn(hObject, eventdata, handles)
% hObject    handle to std_channel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
    get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function avg_xforce_Callback(hObject, eventdata, handles)
% hObject    handle to avg_xforce (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of avg_xforce as text
%        str2double(get(hObject,'String')) returns contents of avg_xforce as a double

% --- Executes during object creation, after setting all properties.
function avg_xforce_CreateFcn(hObject, eventdata, handles)
% hObject    handle to avg_xforce (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
    get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function std_xforce_Callback(hObject, eventdata, handles)
% hObject    handle to std_xforce (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of std_xforce as text
%        str2double(get(hObject,'String')) returns contents of std_xforce as a double

% --- Executes during object creation, after setting all properties.
function std_xforce_CreateFcn(hObject, eventdata, handles)
% hObject    handle to std_xforce (see GCBO)
APPENDIX B (continued)

% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
   get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

function sens_cutting_Callback(hObject, eventdata, handles)
% hObject handle to sens_cutting (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of sens_cutting as text
%       str2double(get(hObject,'String')) returns contents of sens_cutting as a double

% --- Executes during object creation, after setting all properties.
function sens_cutting_CreateFcn(hObject, eventdata, handles)
% hObject handle to sens_cutting (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
   get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

function sens_thrust_Callback(hObject, eventdata, handles)
% hObject handle to sens_thrust (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of sens_thrust as text
%       str2double(get(hObject,'String')) returns contents of sens_thrust as a double

% --- Executes during object creation, after setting all properties.
function sens_thrust_CreateFcn(hObject, eventdata, handles)
% hObject handle to sens_thrust (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles empty - handles not created until after all CreateFcns called
APPENDIX B (continued)

% Hint: edit controls usually have a white background on Windows. 
% See ISPC and COMPUTER. 
if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white'); 
end

function sens_X_Callback(hObject, eventdata, handles)
% hObject    handle to sens_X (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of sens_X as text 
%        str2double(get(hObject,'String')) returns contents of sens_X as a double

% --- Executes during object creation, after setting all properties. 
function sens_X_CreateFcn(hObject, eventdata, handles)
% hObject    handle to sens_X (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows. 
%       See ISPC and COMPUTER. 
if ispc && isequal(get(hObject, 'BackgroundColor'), get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white'); 
end

function sens_channel_Callback(hObject, eventdata, handles)
% hObject    handle to sens_channel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of sens_channel as text 
%        str2double(get(hObject,'String')) returns contents of sens_channel as a double

% --- Executes during object creation, after setting all properties. 
function sens_channel_CreateFcn(hObject, eventdata, handles)
% hObject    handle to sens_channel (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows. 
%       See ISPC and COMPUTER.
function freq_Callback(hObject, eventdata, handles)
    % hObject    handle to freq (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)

    % Hints: get(hObject,'String') returns contents of freq as text
    %        str2double(get(hObject,'String')) returns contents of freq as a
double

% --- Executes during object creation, after setting all properties.
function freq_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to freq (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    empty - handles not created until after all CreateFcns called

    % Hint: edit controls usually have a white background on Windows.
    %       See ISPC and COMPUTER.
    if ispc && isequal(get(hObject,'BackgroundColor'),
        get(0,'defaultUicontrolBackgroundColor'))
        set(hObject,'BackgroundColor','white');
    end

% --- Executes on button press in browse_file.
function browse_file_Callback(hObject, eventdata, handles)
    % hObject    handle to browse_file (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)

    global FileName FilePath Force_Data Name_Cell
    [FileName,FilePath] = uigetfile('.xlsx','Select TIF file');
    set(handles.File_path,'String',FilePath,'horizontalAlignment','right')%To see the Path
    set(handles.File_name,'String',FileName,'horizontalAlignment','left')%To see the Name of The file

% --- Executes during object creation, after setting all properties.
function avg_chan_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to avg_chan (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    empty - handles not created until after all CreateFcns called

    % Hint: edit controls usually have a white background on Windows.
    %       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
    get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in reset_all.
function reset_all_Callback(hObject, eventdata, handles)
% hObject    handle to reset_all (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global zeero yslide

set(handles.y_check,'Enable','on');
set(handles.z_check,'Enable','on');
set(handles.x_check,'Enable','on');
set(handles.channel_check,'Enable','on');
set(handles.y_posi,'string',0);

axes(handles.cutting) cla reset;
axes(handles.cutting_fft) cla reset;
axes(handles.thrust) cla reset;
axes(handles.thrust_fft) cla reset;
axes(handles.fricction) cla reset;
axes(handles.X_fft) cla reset;
axes(handles.channel_fft) cla reset;
axes(handles.channel) cla reset;
axes(handles.Plot_graph) cla reset;

set(handles.avg_cuttingforce, 'String','');
set(handles.avg_thrustforce, 'String','');
set(handles.avg_xforce, 'String','');
set(handles.avg_fricction, 'String','');
set(handles.avg_chan, 'String','');

set(handles.std_cuttingforce, 'String','');
set(handles.std_thrustforce, 'String','');
set(handles.std_xforce, 'String','');
set(handles.std_fricction, 'String','');
set(handles.std_chan, 'String','');

% Force_OpeningFcn(hObject, eventdata, handles, varargin);
clearvars;
yslide = 500;
function rakeangle_Callback(hObject, eventdata, handles)
% hObject    handle to rakeangle (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of rakeangle as text
%        str2double(get(hObject,'String')) returns contents of rakeangle as a double

% --- Executes during object creation, after setting all properties.
function rakeangle_CreateFcn(hObject, eventdata, handles)
% hObject    handle to rakeangle (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
% See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on slider movement.
function Y_slide_Callback(hObject, eventdata, handles)
% hObject    handle to Y_slide (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
global yslide

    yslide = get(handles.Y_slide,'Value');
    set(handles.slidevalue,'String',num2str(yslide));
    plotsep_Callback(hObject, eventdata, handles);

% --- Executes during object creation, after setting all properties.
function Y_slide_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Y_slide (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
APPENDIX B (continued)

```matlab
% --- Executes on button press in Save_plots.
function Save_plots_Callback(hObject, eventdata, handles)
% hObject    handle to Save_plots (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global yslide FileName FilePath dir Time_crop Y_crop X_crop Z_crop mu x3 y3 xtox ylp ylp2 ext y_position
global Z Z_force Y_force Y Fs L P1 P2 f YorN Y_avg X_avg Z_avg Y_std X_std Z_std mu_std mu_avg Yli

dir = [FilePath FileName(1:(length(FileName)-5))];
YorN = exist(dir);
if YorN ~= 7
    mkdir(dir);
end
dirfile_name = [dir ' \ FileName(1:(length(FileName)-5)) '_cutting.jpg'];
figure(1);
axis([-inf inf -yslide yslide]);
axl = gca;
axl.YAxisLocation = 'origin';
axl.XAxisLocation = 'origin';
title('Cutting force', 'FontSize', 16);
xlabel('Time (s)', 'FontSize', 16);
ylabel('Cutting Force (N)', 'FontSize', 16);
set(gca, 'YTick', [-yslide : yslide/6 : yslide]);
cla;
hold on;
plot(Time_crop, Y_crop);
hold off;
saveas(gcf, dirfile_name);
close

figure (2);
dirfile_name = [dir ' \ FileName(1:(length(FileName)-5)) '_thrust.jpg'];
axis([-inf inf -yslide yslide]);
ax2 = gca;
ax2.YAxisLocation = 'origin';
ax2.XAxisLocation = 'origin';
title('Thrust force', 'FontSize', 16);
xlabel('Time (s)', 'FontSize', 16);
ylabel('Thrust Force (N)', 'FontSize', 16);
set(gca, 'YTick', [-yslide : yslide/6 : yslide]);
cla;
hold on;
plot(Time_crop, Z_crop, 'red');
hold off;
saveas(gcf, dirfile_name);
close

figure (3);
dirfile_name = [dir ' \ FileName(1:(length(FileName)-5)) '_coeffriction.jpg'];
axis([Time_crop(xtox) Time_crop(size(Time_crop,1)-xtox) -5 5]);
ax3 = gca;
```

109
ax3.YAxisLocation = 'origin';
ax3.XAxisLocation = 'origin';

ylh = get(gca,'xlabel');
gyl = get(ylh);

% Object Information
ylp = get(ylh,'Position');
ext = get(ylh,'Extent');
set(ylh,'Rotation',0,'Position',ylp+[0 -1.5 0],
'VerticalAlignment','middle','HorizontalAlignment','right');

% title('Coefficient of Friction');
xlabel('Time (s)', 'FontSize',16);
ylabel('Coefficient of Friction', 'FontSize',16);
set(gca,'YTick',[-5:1:5]);
cla;
hold on;
plot(Time_crop,mu,'green');
fric_Y = [0,1,2,3];
plot((x3(1)*ones(1,4)),fric_Y,'--k');
plot((x3(2)*ones(1,4)),fric_Y,'--k');
s1 = ['Avg. Coefficient of friction= ' num2str(round(mu_avg,2)) ' ± ' num2str(round(mu_std,2))];
s2 = ['Time limits for average'];
lgd = legend(s1,s2,'Location','southeast');
lgd.FontSize = 14;
hold off;
saveas(gcf,dirfile_name);
close

Z = fft(Z_force);
L = size(Z_force,1);
P2 = abs(Z/L);
P1 = P2(1:L/2+1);
P1(2:end-1) = 2*P1(2:end-1);
f = (Fs*(0:(L/2))/L);
figure(4);
dirfile_name = [dir '\' FileName(1:(length(FileName)-5)) '_FFT_thrust.jpg'];
plot(f,P1);
%axis ([0 10000 0 1000]);
title('Single-Sided Amplitude Spectrum of Thrust froce');
xlabel('f (Hz)', 'FontSize',16);
ylabel('Amplitude', 'FontSize',16);
saveas(gcf,dirfile_name);
close

Y = fft(Y_force);
L = size(Y_force,1);
P2 = abs(Y/L);
P1 = P2(1:L/2+1);
P1(2:end-1) = 2*P1(2:end-1);
f = (Fs*(0:(L/2))/L)';
figure (5);
dirfile_name = [dir ' \ FileName(1:(length(FileName)-5)) '_FFT_cutting.jpg'];
plot(f,P1);
axis ([0 1000 0 1000]);
title('Single-Sided Amplitude Spectrum of Cutting force');
xlabel('f (Hz)', 'FontSize',16);
ylabel('Amplitude', 'FontSize',16);
saveas(gcf,dirfile_name);
close

y_position = str2num(char(get(handles.y_posi, 'String')));
figure (6);
dirfile_name = [dir ' \ FileName(1:(length(FileName)-5)) '_cutting&thrust.jpg'];
axis([-inf inf 0 yslide]);
ax2 = gca;
ax2.YAxisLocation = 'origin';
% ax2.YLabel.String = 'Force (N)';
ax2.XAxisLocation = 'origin';
ylmh = get(gca,'ylabel');
ylp2 = get(ylmh,'Position');
ext=get(ylmh,'Extent');
set(ylmh,'Rotation',90,'Position',[y_position 300 0],'
VerticalAlignment','middle', 'Horizontal_ALIGNMENT','center');

% title('Machining force');
dim = [0.2 0.5 0.3 0.3];
% annotation('textbox',dim,'String',str,'FitBoxToText','on');
xlabel('Time (s)', 'FontSize',16);
ylabel('Force (N)', 'FontSize',16);
set(gca,'YTick',[0 : 100 : yslide]);
set(gca, 'FontSize', 16);
Ylimshow = [-100:1:((4*yslide/5)-100)];
cla;
hold on;
plot(Time_crop,Y_crop,'Color',[1 0.5468 0]);
plot(Time_crop,Z_crop, 'green');
plot((x3(1)*ones(1,((4*yslide/5)+1))),Ylimshow,'--k');
plot((x3(2)*ones(1,((4*yslide/5)+1))),Ylimshow,'--k');
dim = [0.5 0.5 0.3 0.3];

sA = 'Average';
s5 = ['Avg. Cutting Force (N)= ' num2str(round(Y_avg,0)) ' ± ' num2str(round(Y_std,0))];
s10 = ['Avg. Thrust Force (N)= ' num2str(round(Z_avg,0)) ' ± ' num2str(round(Z_std,0))];
s11 = ['Time limits for average'];
% a = annotation('textbox',dim,'String',{sA,s5,s10},'FitBoxToText','on');
% a.FontSize = 16;
lgd = legend(s5,s10,s11,'Location','northeast');
lgd.FontSize = 14;
APPENDIX B (continued)

hold off;

saveas(gcf,dirfile_name);
close

zero = 0;
what_force = {'Cutting Force';'Thrust force';'Coefficient of Friction';'Time start for average';'Time end for average'};
Average = [round(Y_avg,0); round(Z_avg,0); round(mu_avg,2);x3(1);x3(2)];
Std_dev = [round(Y_std,0); round(Z_std,0); round(mu_std,2);zero;zero];

T = table(Average, Std_dev,'RowNames',what_force);
dirfile_name = [dir '\' FileName(1:(length(FileName)-5)) '_statistics.csv'];
dirfile_name2 = [dir '\' FileName(1:(length(FileName)-5)) '_statistics.txt'];
writetable(T,dirfile_name,'WriteRowNames',true);
writetable(T,dirfile_name2,'WriteRowNames',true);

function y_posi_Callback(hObject, eventdata, handles)
    % hObject    handle to y_posi (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    
    % Hints: get(hObject,'String') returns contents of y_posi as text
    %        str2double(get(hObject,'String')) returns contents of y_posi as a double

    % --- Executes during object creation, after setting all properties.
    function y_posi_CreateFcn(hObject, eventdata, handles)
    % hObject    handle to y_posi (see GCBO)
    % eventdata  reserved - to be defined in a future version of MATLAB
    % handles    structure with handles and user data (see GUIDATA)
    
    % Hints: edit controls usually have a white background on Windows.
    %        See ISPC and COMPUTER.
    if ispc && isequal(get(hObject,'BackgroundColor'),
        get(0,'defaultUicontrolBackgroundColor'))
        set(hObject,'BackgroundColor','white');
    end

    % --- Executes on button press in Trig.

112
## APPENDIX C

Cast wrought IN 625 material certificate

---

### Certificate of Compliance

<table>
<thead>
<tr>
<th>Line</th>
<th>Product Description</th>
<th>Ordered</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8786K3 High-Strength 625 Nickel Sheet with Material Certification, 0.040&quot; Thick, 6&quot; x 6&quot;</td>
<td>1 Each</td>
</tr>
<tr>
<td>2</td>
<td>8731K115 High-Strength 625 Nickel Rod with Material Certification, 1&quot; Diameter, 6&quot; Long</td>
<td>1 Each</td>
</tr>
<tr>
<td>3</td>
<td>1773T2 High-Strength 17-4 PH Stainless Steel Sheet 0.040&quot; Thick, 6&quot; x 6&quot;</td>
<td>1 Each</td>
</tr>
<tr>
<td>4</td>
<td>1319T1 High-Strength 17-4 PH Stainless Steel Rod 3&quot; Diameter, 3&quot; Long</td>
<td>1 Each</td>
</tr>
<tr>
<td>5</td>
<td>90251A549 Thread Locking Cup Point Set Screw 18-8 Stainless Steel, 1/4&quot;-20 Thread, 3/4&quot; Length, Packs of 10</td>
<td>2 Packs</td>
</tr>
<tr>
<td>6</td>
<td>90251A537 Thread Locking Cup Point Set Screw 18-8 Stainless Steel, 1/4&quot;-20 Thread, 1/2&quot; Length, Packs of 10</td>
<td>2 Packs</td>
</tr>
<tr>
<td>7</td>
<td>9018K11 Multipurpose O1 Tool Steel Flat Stock, 3/16&quot; x 1/2&quot;, 1/2&quot; Long</td>
<td>1 Each</td>
</tr>
<tr>
<td>8</td>
<td>9018K21 Multipurpose O1 Tool Steel Flat Stock, 1/4&quot; x 1/2&quot;, 1/2&quot; Long</td>
<td>1 Each</td>
</tr>
</tbody>
</table>
## APPENDIX C (continued)

![Image of material certification page](image-url)

<table>
<thead>
<tr>
<th>Customer Order No.</th>
<th>Order Date</th>
<th>Cert Date</th>
<th>Shop Order</th>
<th>Alloy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0112016-WIN #2</td>
<td>05/05/2014</td>
<td>11/20/2014</td>
<td>P77562-1</td>
<td>625</td>
</tr>
</tbody>
</table>

### Specifications:
- AMS 56655, ASTM E-446-03 (2008) El Gr. 1, ASME SB-446-2013 Ed. Gr. 1,
- S400 02/21/14, S-1000 12/04/13, SS0500: B80e, ASGR-01 REV. 8,

### Quantity (Pcs./Wt.):
- 2 BOLLS / 3419 LBS

### Size of Dwg. Number:
- 1.000" RD. X 10" - 14" R/L

### Heat Number:
- 192419

### Material Analysis:

<table>
<thead>
<tr>
<th>Element</th>
<th>%</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td></td>
<td>0.034</td>
</tr>
<tr>
<td>Mn</td>
<td></td>
<td>0.19</td>
</tr>
<tr>
<td>Si</td>
<td></td>
<td>0.214</td>
</tr>
<tr>
<td>P</td>
<td></td>
<td>0.005</td>
</tr>
<tr>
<td>S</td>
<td></td>
<td>0.001</td>
</tr>
<tr>
<td>Cr</td>
<td></td>
<td>22.87</td>
</tr>
<tr>
<td>Ni</td>
<td></td>
<td>60.64</td>
</tr>
<tr>
<td>Co</td>
<td></td>
<td>0.05</td>
</tr>
<tr>
<td>Mo</td>
<td></td>
<td>8.85</td>
</tr>
<tr>
<td>Nb + Ta</td>
<td></td>
<td>3.49</td>
</tr>
</tbody>
</table>

### Material Source:
- Melted at VDM Metals USA/PP

### Mechanical and Metallurgical Tests:

<table>
<thead>
<tr>
<th>Test</th>
<th>Condition</th>
<th>Temp. (F)</th>
<th>YS (ksi)</th>
<th>UTS (ksi)</th>
<th>%El. (4D)</th>
<th>%R.A.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>RT (LONG)</td>
<td>141.0</td>
<td>75.5</td>
<td>52.0</td>
<td>60.0</td>
<td></td>
</tr>
</tbody>
</table>

### Certification:

- **Rolling Alloys: Quality Assurance**
- **Approver: Donald Artieri / Senior Quality Supervisor**
- **Date: 1/1/14**

---

**Material Description**

- **Material:** VDM Metals USA, LLC
- **Address:** 125 West Green Road, Temperance, MI, USA
- **Phone:** 734-972-6000

---

**Material:** VDM Metals USA, LLC

- **Address:** 125 West Green Road, Temperance, MI, USA
- **Phone:** 734-972-6000
CERTIFIED MATERIALS TEST REPORT

RA TRACER # 0485150 US

Material Description
INCONEL alloy 625, SAF-600 ELECTROSLAG REMELTED, CR SHEET IN COIL, COLD ROLLED, ANNANLED, 0.0405 X 36,0000G, IN THICKNESS TOLERANCE +/- 0.002
1 PC 6114 LBS

Specifications
SAE AMS 5599G MARKING WAIVED/ ASME SB-443 2015 EDITION GR 1/ ASTM B 443-00 (2014) GR 1/
HONEYWELL AMS 95377 KRV D MARKING WAIVED/ GE S-800 (S-18-15)/ GE S-1000 (S-22-10)/ WORKSTAFF 2.4856/ NACE

RA TRACER # 0485150 US

ANALYSIS

<table>
<thead>
<tr>
<th>Element</th>
<th>C %</th>
<th>Fe %</th>
<th>Ni %</th>
<th>Si %</th>
<th>Ni %</th>
<th>Cr %</th>
<th>Al %</th>
<th>Ti %</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>.02</td>
<td>.07</td>
<td>3.85</td>
<td>.001</td>
<td>.15</td>
<td>60.5</td>
<td>22.35</td>
<td>.33</td>
</tr>
<tr>
<td>Method</td>
<td>C/S</td>
<td>ICP</td>
<td>ICP</td>
<td>C/S</td>
<td>ICP</td>
<td>ICP</td>
<td>ICP</td>
<td>ICP</td>
</tr>
</tbody>
</table>

INCOLOY®, INCONEL®, MONEL®, Nilo®, NIVONIC®, Ni-Span®, Uranmit®, Duramack®, 601GC®, 625MC®,
are trademarks of the Special Metals group of companies.

EP595
HUNTINGTON ALLOYS CORPORATION
2109 Riverside Drive, Huntington, West Virginia 25701-1714 USA
Tel: +1.304.626.0100  Toll-Free in the USA: 1.800.334.4626
Fax: +1.304.626.9543  info@specialmetals.com

CERTIFICATE NO. N63978-00

CERTIFIED MATERIALS TEST REPORT

<table>
<thead>
<tr>
<th>CO %</th>
<th>NO %</th>
<th>NS %</th>
<th>TA %</th>
<th>P %</th>
</tr>
</thead>
<tbody>
<tr>
<td>.12</td>
<td>.68</td>
<td>3.47</td>
<td>.004</td>
<td>.008</td>
</tr>
<tr>
<td>Method</td>
<td>ICP</td>
<td>ICP</td>
<td>ICP</td>
<td>ICP</td>
</tr>
</tbody>
</table>

SALTAR

<table>
<thead>
<tr>
<th>H</th>
<th>3.07</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td></td>
</tr>
</tbody>
</table>

ANALYSIS METHOD LEGEND

ICP - Inductively Coupled Plasma
C/S - Carbon/Sulfur

TENSILE TEST

ROOM TEMPT TENSILE - TRANS/LAB/MECHANCIAL

test temp hardnes hard type tensile ksi yield ksi eff ga length in long4 orient

<table>
<thead>
<tr>
<th>AN</th>
<th>157</th>
<th>135.5</th>
<th>70.4</th>
<th>2</th>
<th>52.2</th>
<th>TENS</th>
</tr>
</thead>
</table>

HEAT TREATMENT

<table>
<thead>
<tr>
<th>TEMP</th>
<th>FURNACE</th>
<th>TEMP SCALE</th>
<th>TEMP 1 F</th>
<th>TEM 1 MOLD HOLES</th>
<th>UNITS</th>
<th>CHARGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAL</td>
<td>F</td>
<td>1000</td>
<td>1.4</td>
<td>MIN</td>
<td>AC</td>
<td>1</td>
</tr>
</tbody>
</table>

OTHER TESTS

BEND TEST - TRANS/LAB/MECHANCIAL

test tempret test verdict bend factor bend angle crack check orient

<table>
<thead>
<tr>
<th>AN</th>
<th>F</th>
<th>15</th>
<th>180</th>
<th>NC</th>
<th>TENS</th>
</tr>
</thead>
</table>

GRAIN SIZE MEASUREMENT/LAB/METALLOGRAPHY

test tempret test orient av go astm hrn or dup

<table>
<thead>
<tr>
<th>AN</th>
<th>TRANSVERSE</th>
<th>8.5</th>
<th>NORMAL</th>
</tr>
</thead>
</table>

RA TRACER # 048S15O_US
APPENDIX C (continued)

CERTIFIED MATERIALS TEST REPORT

Dated: 21-JUL-16

Page No: 3 / 4

NO WELD REPAIR HAS BEEN PERFORMED ON THIS MATERIAL.

LOCATION LEGEND: B = BASE  C = CENTER  F = FRONT  H = HEAD  M = MIDDLE  T = TOR

TEST VERDICT LEGEND: F = PASS  W = WAIVER

TEST TEMPER LEGEND
AS = Annealed

ALL TEST RESULTS ARE REPORTED TO AT LEAST THE REQUIRED PRECISION BY THE ROUNDED METHOD OF ASTM E 29
UNLESS OTHERWISE REQUIRED BY PURCHASE ORDER OR SPECIFICATION.

COUNTRY OF ORIGIN: WELDED AND MANUFACTURED IN THE USA. OPEARS PART 252.225-7014 AND 252.225-7008 COMPLIANT.

THIS CERTIFICATION AFFIRMS THAT THE CONTENTS OF THIS REPORT ARE CORRECT AND ACCURATE AND THAT ALL TEST RESULTS AND OPERATIONS PERFORMED BY SPECIAL METALS CORPORATION, INC. OR ITS SUBCONTRACTORS ARE IN COMPLIANCE WITH THE MATERIAL SPECIFICATIONS.

QUALITY SYSTEM MEETS REQUIREMENTS OF DIRECTIVE 97-23/EC (PRESSURE EQUIPMENT DIRECTIVE), ANNEX 1. CHAPTER 4.3 PBRA AGB LTD CERTIFICATE 41734 (EXPIRES JULY 28, 2017)

HUNTINGTON ALLOYS CORPORATION IS AN ACCREDITED INDEPENDENT NONDESTRUCTIVE MATERIALS TESTING LABORATORY VIA CERTIFICATION NUMBER 3060087834 (REVISED APRIL 29, 2016) FOR ALL TESTING SPECIFIED IN THE SCOPE OF ACCREDITATION.

MATERIAL TESTING LABORATORY COMPLIES WITH QA SYSTEM DOCUMENTED IN HUNTINGTON ALLOYS CORP QA MANUAL REV. B, DATED 10/17/2014

QA MANUAL NOT TO IMPY CONFORMANCE TO ASME SECTION III. COMPLIANCE MUST BE OTHERWISE STATED ON THE CERTIFICATE.

QUALITY SYSTEM CERTIFICATION: ISO 9001:2008 (ASD-QS-9000, NQA1), EN 10-204/2002 10049 (TYPE 3.1)

LABORATORY IS ACCREDITED TO ISO 17025:2005 FOR MECHANICAL TESTING AND CHEMICAL ANALYSIS.

VISUAL AND DIMENSIONAL EXAMINATION SATISFACTORY.

MATERIAL, WHEN FITTED, IS FREE FROM CONTAMINATION BY MERCURY, RADIUM, ALBÉRA SOURCES, AND LOW MELTING ELEMENTS.

CHEMICAL ANALYSIS AS REQUIRED FOR CARBON, SULFUR, NITRGEN, OR OXYGEN IS PERFORMED BY COMBUSTION TECHNIQUES.

ALL OTHER REPORTED ELEMENTS ARE ANALYZED BY X-RAY RAY AND/OR EMISSION SPECTROSCOPY.

AUTHORIZED QUALITY CERTIFICATION REPRESENTATIVES:


GE Transportation Aviation Special Process Certification Source Code 47150, Expiration Date of 7/31/2016

INCOLOY®, INCOSEL®, MONE®, NICO®, NIMONIC®, N-S-SPAN®, UDÉM©, DURANIC®, 601DC®, 831CF®, 718PH®, 740H®, 800H®, 986®, and 946® are trademarks of the Special Metals Group of companies

RA TRACER #: 946515C US

117
This is to certify that all required sampling inspections and tests have been performed in accordance with the order and specification requirements. The test report represents the actual attributes of the material furnished and the values shown are correct and true. The material described by this certificate is in full compliance with all order and inspection requirements. We hereby certify that the figures given are in accordance with the specified contract requirements.
APPENDIX C (continued)

October 27, 2016

Re: RECERTIFICATION

Customer: Rolled Alloys, Inc.

This document is to certify that INCONEL alloy 625 sheet and plate supplied to Rolled Alloys, Inc. by SMC Huntington, WV and certified to SAE AMS 5599G also meets the requirements of SAE AMS 5599H except marking and with the following clarification on plate greater than 1.000 inch in nominal thickness: the Certified Material Test Report will not have a statement that the product size is outside the range covered by this specification.

This letter expires April 1, 2018 due to the requirement for compliance with AS6279 that is effective on that date.

The above information is true and accurate as certified by the person stated below.

[Signature]

Deb Fendle
Supervisor Specifications
SMC Huntington

[Stamp]
ROLLED ALLOYS QUALITY ASSURANCE
APPROVED
DATE 10/5/16