2023-12-01

Exploration Of Advanced Scan Strategy Capabilities On A Commercial LPBF System Using Intervector Power Ramping

Antonio York
University of Texas at El Paso

EXPLORATION OF ADVANCED SCAN STRATEGY CAPABILITIES

ON A COMMERCIAL LPBF SYSTEM USING

INTERVECTOR POWER RAMPING

ANTONIO JAMES YORK

Master's Program in Mechanical Engineering

APPROVED:

David Espalin, Ph.D., Chair

Ryan Wicker, Ph.D.

Amit Lopes, Ph.D.

Stephen L. Crites, Jr., Ph.D.
Dean of the Graduate School

DEDICATION

I hereby dedicate this body of work to my family and friends. Without their support I have no doubt I would not have completed this program nor sought knowledge in this field in the first place. To my parents, thank you for instilling in me the value of hard work. To my brothers, thank you for having a laugh with me whenever things were rough. To the guys from Tailored Alloys, thanks for the good times, and for providing opportunities for me to try new things. Robby, my fellow international businessman, LTX was a blast, thanks man! Jason thanks for the ramen that one time and your copy of Diablo 2…or maybe not…it's actually really challenging. Lastly thanks to every colleague that helped keep me sane these past couple years.

EXPLORATION OF ADVANCED SCAN STRATEGY CAPABILITIES ON A COMMERCIAL LPBF SYSTEM USING INTERVECTOR POWER RAMPING

by

ANTONIO JAMES YORK, B.S.M.E.

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at El Paso

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

Department of Aerospace & Mechanical Engineering

THE UNIVERSITY OF TEXAS AT EL PASO

December 2023

ACKNOWLEDGEMENTS

I'd like to express my gratitude to my fellow researchers at the W.M. Keck Center who were always available for an assist and to listen to me ramble like a madman. Thanks to staff members Mahesh Tonde and Mireya Flores for always helping me in a jiffy. Additional appreciation goes out to Jose Coronel for helping me navigate the odd situation, Dakota Morgan for always being around and helping me see things from different perspectives, and especially Jorge Mireles and Dr. Wicker for surviving all of my updates. I'm also extremely grateful to Dr. Espalin for all his time spent mentoring and challenging me, I've still got a ways to go but I still aim to exceed your expectations. Lastly, a tip of the hat to H-Bomb, no doubt a guy with the drive to accomplish something really cool no matter what setback comes his way, thanks for motivating me. This material is based on research sponsored by Air Force Research Laboratory under agreement number FA8650-20-2-5700. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory or the U.S. Government.

ABSTRACT

Lack of confidence in the quality of materials produced with metal AM has hinged on the unaddressed defects inherent to the process. Studies have worked to determine defect formation mechanisms and provide adjustments to key process controls. However, these advances have typically been developed in-house on research systems which offer increased and transparent control to the operator. A research gap exists in applying changes in approach of scan strategies to existing commercial systems, which can incentivize adoption while furthering development. The focus of this study is the implementation of inter-vector power ramping on a commercial laser powder bed fusion system to mitigate variability of melt-pool geometry in transient states. System specific kinematics were characterized and an analytical model based on normalized enthalpy was simplified to calculate power adjustments. Scanner encoder position data was monitored and used to derive velocity and acceleration for the test geometry. MATALB was used for data processing and to develop a nonlinear velocity approximation. A python script was developed to generate and visualize system compatible build files. Initial tests revealed inaccurate spatial/temporal positioning of the applied ramp, requiring modified power adjustment vector indexes to offset the ramp. Single layer plate scans were performed comparing constant and ramped power when accelerating. Laser and scanner monitoring data and surface measurements were used to verify scan strategy execution while depth measurements from micrographs of melt-pool cross-sections were used to quantify the ramp effectiveness. Initial results proved useful in identifying several limitations in data processing and the effects of hardware constraints on implementing functionality for advanced LPBF process control.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1 Background

Although laser power control has been demonstrated using real-time control, feedforward, and feedback approaches, it requires development of research platforms to integrate additional hardware. Rather than develop proving grounds for cutting-edge LPBF control mechanisms or simulations, this study utilizes an approach which implements existing power control on a commercial system. With the selected approach the requirements for defining ramps are simplified but the result does not offer real-time control.

The laser powder bed fusion (LPBF) additive manufacturing (AM) process uses a high-power laser to, "scan", melt selective regions of a layer of metal powder. Multiple layers, 2D cross-sections, are repeated producing a 3D part. This requires several layers of data processing: producing CAD geometry, STL conversion or tessellation of geometry, nesting, support generation, assignment of parameters, slicing and toolpath generation, and additional parameter assignment (typically at the machine). A build file containing all relevant information is then used to manufacture components which can be post-processed to modify aesthetic, geometric, or mechanical properties. LPBF offers increased control resolutions compared to many other manufacturing methods, with features in the micrometer regime capable of being processed. This also provides an opportunity for anomalous features and process defects to manifest. The type explored in this study stems from limitations on parameter adjustment while processing between and within single vectors. Although this is just one of LPBF's many limitations, presenting solutions to improve the way this technology processes materials can have ramifications on not just how our parts perform but how others view this technology and its potential. The hope of

this student is not that this process solves every manufacturing problem, replacing subtractive and other additive methods, but that this technology matures to a point where the products of it can be disseminated with as much confidence as forged or cast parts.

1.2 Motivation

LPBF offers an extreme level of control in comparison to subtractive and even other AM processes. However, the potential for error with that level of control is just as extreme. While producing intricate components for a variety of industries is already possible, confidence in the process, specifically for functional components requiring certification & qualification, is lacking. Developing deeper understandings of the complex process dynamics that occur is of great personal interest and will provide opportunities for wider adoption of the technology as material quality improves. Although serving as just one tool in today's arsenal of manufacturing options, applications for LPBF extend from medical prosthesis and automotive components to cutting edge alloy development and processing in aerospace industries. Demonstrating improved processing with this specific scan strategy is just one small step towards a large process revolution for increased reliability and performance.

1.3 Thesis Objectives

The goals of this study comprised:

1. Using an existing model to define a power ramp that normalizes end-of-vector depth.
2. Exploring methods for adapting a nonlinear power profile on the commercial system.
3. Evaluating performance of ramped power (RP) versus constant power (CP) scanning.

1.4 Thesis Outline

The purpose of this thesis is to document the research discussed herein and serve as a reference for additional studies focused on improving LPBF and similar manufacturing

processes. Chapter 2 includes a review of relevant literature on the LPBF process, advancements in monitoring and modelling for process induced defect identification and prevention, and remarks on current capabilities/limitations of commercial systems. Chapter 3 covers the testing methodology, adjusted power profile definition, and system-specific implementation approach. Chapter 4 presents results in the turnarounds and identifies issues with the execution of the intended scan strategy. Chapter 5 interprets results in the context of ramp effectiveness for reducing depth variability in turnarounds. Chapter 6 provides concluding statements regarding this study and potential future directions of study.

CHAPTER 2: LITERATURE REVIEW

Parts produced by commercial LPBF systems contain defects including: residual stresses, keyholes, lack of fusion, balling, etc. [1]. A key factor in producing these defects is the variability in operation of the laser or scanner, and the limited control over a component's thermal history within and between layers. The need for modern, advanced control has been documented in [2] in which the national institute of standards and technology (NIST) outlined the development of their AM testbed to better characterize the LPBF process in addition to proposing predictive, in-process monitoring, and simulation-based control strategies. Along with development of open-architecture LPBF systems, progress towards build preparation software offering increased transparency to operators may help to reduce variability of part production between different systems while also adding functionality such as the point-wise parameter interpolation for higher resolution control [3]. A variety of analysis methods for monitoring and predicting process defects/physics, performing system diagnostics, and enhancing part properties have been advancing and are discussed to review the state-of-the-art as well as current limitations some of which were the basis for the approach taken during experimentation. A simple but effective “bottom up” approach widely used in efforts to optimize parameters and scan strategies includes the use of single scan tracks and subsequent analysis of surface morphology and melt-pool cross-sections. Improved densification of parts has been documented in [4] where an automated analysis process of track surfaces was used to characterize process windows for multiple materials on multiple systems.

It has been shown that systems lacking a feedback mechanism are less capable at maintaining consistent input energy density to the melt pool [5]. In the study a NIR camera was

compared to a photodiode to account for the spatial distribution of temperature in the melt pool rather than average temperature highlighting the complex thermal histories components in the LPBF process undergo and the need for advanced laser power controllers to modulate power not just between but within vectors to limit temperature variations during manufacturing [6-7]. These controllers may also benefit laser scanning profiles [8], so that rather than approximating a nonlinear system response and applying a case-specific profile, feedback into the profile can regulate it in real-time. Highlighting the need for reducing thermal deviations during transient events at ends of vectors are the advances in the use of instrumentation to measure characteristics of the LPBF process such as the use of optical tomography and infrared thermography to correlate temperature deviations with defects [9]. Additionally, thermal gradients and cooling rates have been measured and found to be dependent on geometry such as ends of vectors as well as process parameters [10].

Using correlation analysis to identify process parameters and material properties with highest impacts on melt-pool phenomena Lee et al. determined that machine learning algorithms could be employed to accurately predict melt-pool geometry in the substrate region [11]. Using a feed-forward control scheme, researchers reduced melt-pool image area by 78% and seek expansion of experimental training datasets to enhance model accuracy and reduce prediction time [12]. Zhirnov et al. proposed diagnostic methods for determining the cause of single scan track anomalies, such as track shape and straightness, including various single layer plate scans which could be used to troubleshoot a printer's optical system [13]. While some process limitations including energy delivery and assembly defects are constrained by the hardware in use [14-16] other areas including the development of laser profiles for expanding power control or scan strategies can benefit from improving process control and the inclusion of advanced

feedback methodologies [8]. Other studies have considered the benefits of unique raster geometries to benefit from reheating cycles [17] and constant scanning throughout geometry [18] which was utilized in this study.

The use of an off-axis camera setup in observing vapor plumes from a melt pool realized the need for real time process control [19] as in [5], and the results point to the possibility of a reduction in accuracy for predictive models depending on the complexity of a manufactured parts thermal history. Additionally, as seen in both [5] and [20] instrumentation was used to account for the spatial distribution of temperature in the melt pool rather than average temperature highlighting the complex thermal histories components in the LPBF process undergo. High speed x-ray imaging was used to characterize physical processes of LPBF including melt pool development and keyhole formation [21]. The development of open-source data registration methods for camera-based melt pool monitoring have resulted in the alignment of melt pool images with monitoring data allowing for correlation between reconstructed 3D models and the original CAD or post-inspection results [22]. These studies have shown that the acquired data may be used for defect prevention and reduction evolving towards the development of closed loop feedback controllers for improving produced part properties and expansion of LPBF capabilities.

Approaches used in literature for determining the dynamics of galvanometer scanners are limited but include the direct input of signals of varying frequencies to the scanner and measurement of the amplitude and phase delay [8]. Yeung et al. used a sinusoidal input, and were able to verify the calculated acceleration magnitude and time delay with traditional scan tests. Additional studies at NIST considered assembly defects of the optical train and encoder compensation to improve positional accuracy. Other works have relied on the standard form of

inertia compensation (skywriting) which does not resolve the formation of depressions at ends of vectors. Khairallah et. al recommended use of power ramping at ends of vectors to prevent the rapid solidification of material and potential pore formation [23]. The defect, known as a "frozen depression", has been mitigated by synchronizing ramps of both power and scan speed on a custom, research system [24] and a similar approach has been used to mitigate potential formation of end of vector porosity caused by scanner acceleration at corners and turnarounds [25]. The use of determining scanner dynamics extends beyond this particular effort. Although this work focuses exclusively on mitigating end of vector depressions, the use of known kinematic values could be of interest if using modern state-of-the-art controllers. Acceleration-limited or jerk-limited control has been tested as a means of improving scan path positional/temporal accuracy and laser-scanner synchronization by commanding the scanner's motion more closely to its physical limits, rather than relying on delays to compensate for incapabilities [26]. Although traditional controllers and commercial systems lack these capabilities, advancing technology in research systems has enabled fully transparent, synchronized control of the laser and scanner, geometry dependent power, and compensation for dynamic heating during processing [3, 27-28].

Similar research such as parameter prediction based on melt pool imaging [29] or melt pool classification resulting in feedback control of parameters [30], or the development of controls for defect reduction through proximal policy optimization resulting in modulation of either scan velocity or power at powder bed regions where heat accumulates [31], provide examples of advanced controls for the process, however, if the systems on which these enhancements are placed do not possess software with transparent control logic, or do not offer functionality for the definition of parameter relationships beyond what currently exists then some

current limitations will persist. Additional limitations include the lack of available monitoring data on which machine learning algorithms can be trained, which organizations such as [NIST] seek to alleviate [32] and which we seek to add to through the work conducted in this study. While this study focuses on and is constrained by currently accessible monitoring data of a specific printer, integration of additional monitoring capabilities is of great interest such as adjusting power to control temperature and moderate melt-pool surface area [33].

CHAPTER 3: MATERIALS & METHODOLOGY

3.1 Research Materials

An overview of methodology is provided here whereas subsequent sections provide additional details. The study required the use of plate scans and analysis of single scan tracks (SST's). A commercial LPBF printer was used to scan stainless steel plates and optical microscopy was used to analyze the surface and cross-sections of scanned vectors. A precision cutoff saw and grinding/polishing systems were used to prepare the surfaces of plates while chemical etching was performed to reveal the microstructure, highlighting a layer of melt-pools. MATLAB was used to develop tools for data processing, visualization, and calculating approximations of velocity profiles to define nonlinear laser power profiles. Python was used to develop a program that could generate simplified single layer geometry based on user defined parameters. A power adjustment equation for calculating ramps was integrated, geometry could be discretized based on requirements of the build file format, and powers could be assigned to polyline chains enabling functionality lacking in commercial AM software.

3.1.1 System Specifications

Figure 1: Aconity3D MIDI+ LPBF system.

An Aconity MIDI+ (Aconity3D GmbH, Aachen, Germany), Figure 1, printer configured with a nLIGHT (nLIGHT, Vancouver, WA, U.S.) CW, 500W, D4σ ~ 75um, Yb:YAG, 1070nm laser was used for scanning. A SP-ICE-3 (Raylase GmbH, Weßling, Germany) control card commands the laser and the Axialscan Fiber-30 (Raylase GmbH, Weßling, Germany) 3 axis scan

head. The printer was also fitted with an onboard field programmable gate array (FPGA) to acquire data from various sensors. Those important for this review include laser state, laser power, and scanner encoder data. The data is communicated using the SL2-100 (SCANLAB GmbH, Puchheim, Germany) protocol with a sample frequency of 100kHz and stored using Aconity's PCD file format.

3.1.2 Test Samples

To reduce cost associated with iterative testing, scans were performed on prefabricated 304 stainless steel plates (ASTM A240, SAE AMS5513) with a brushed #3 finish which had been cut into ~ 25x25x3mm samples. The plates were cleaned with isopropyl alcohol prior to each scan to prevent contamination and ensure unimpeded energy delivery. Immediately after cleaning, plates were centered on the build platform and the chamber was sealed.

3.1.3 Imaging

Figure 2: Keyence VHX-7000 system.

The VHX-7000 (Keyence, Itasca, IL, U.S.), Figure 2, was used to obtain optical micrographs of the plate surfaces and etched cross-sections. Elements of the scans surface morphology were identified as regions of interest, specifically ends of vectors where track width may increase due to heat accumulation. Images of the surface and section used coaxial lighting, with the E100 optic at 100x magnification for the surface, and 500x for the cross-section. 3D serial recordings (automated image stitching) were used to compensate for sample flatness. The accuracy and repeatability of XY measurements for the optic were 3% of reading and 0.08um respectively. Parallel/perpendicular measurements were taken with reference to a manually set reference line from the first to last object to measure.

3.1.4 Metallography

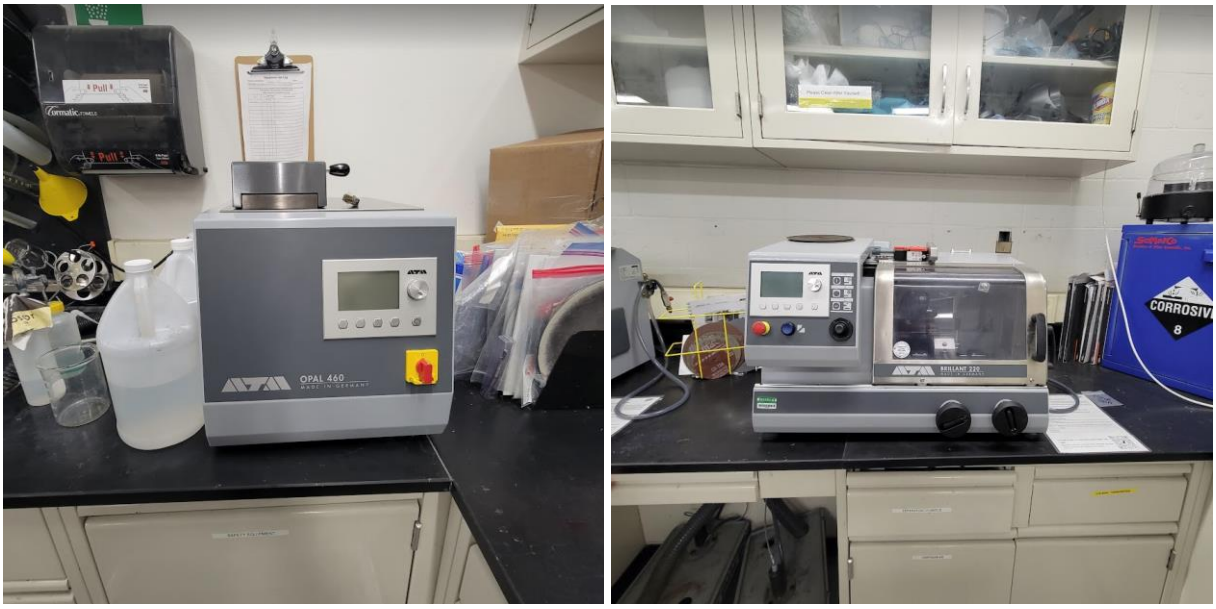

Figure 3: QATM hot press mount and cutoff saw.

Figure 4: QATM Saphir 550 automatic grinding/polishing system.

The plates were sectioned using a Brillant 220 (QATM GmbH, Mammelzen, Germany) cutoff saw and then mounted with an Opal 460 (QATM GmbH, Mammelzen, Germany), Figure 3, hot mount press. A Saphir 550 (QATM GmbH, Mammelzen, Germany), Figure 4, was utilized for grinding and polishing progressively using a 6um diamond suspension, 3um diamond suspension, and 0.1um colloidal silica suspension. Lastly carpenter's etchant was then used to reveal the microstructure. These sections enabled measurement of melt geometry, verification of steady-state parameters, and quantification of the power ramp effectiveness.

3.1.5 Type 1 Geometry

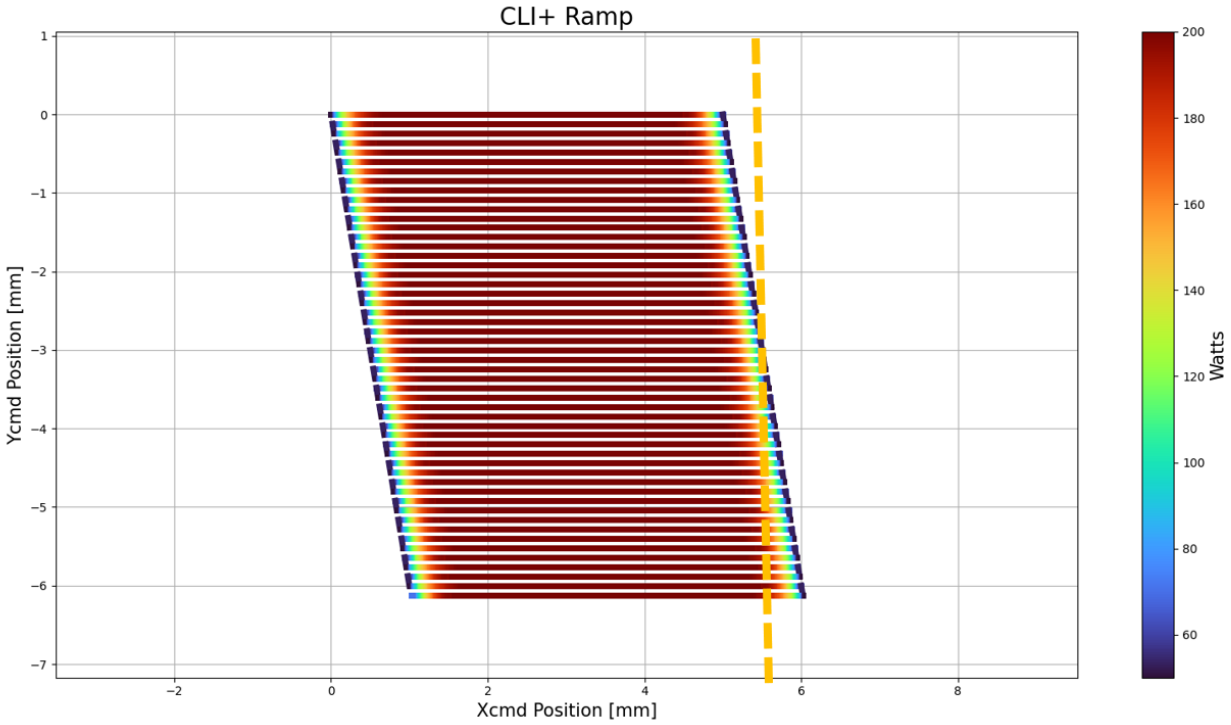

Figure 5: Type 1 geometry sectioning plan.

Additional details regarding differences in geometry and process execution between a standard skywriting mode and the configuration used to achieve testing for type 1 are discussed in section 3.2. Fifty-two vectors each, 5mm in length, with a hatch distance of 120um, were positioned with the first vector's start at the origin of the machine's coordinate system. All

vectors were aligned parallel to the x-axis, parallel to flow in this system, and were scanned with alternating direction (i.e., vector 1 scanned along +x, vector 2 scanned along -x). Additionally, a 20um x-offset was applied between pairs of vectors so that the geometry of melt-pools could be evaluated at multiple distances, with respect to the surface, along the length of the tracks. As seen in Figure 5, if the plate is grinded to a point before the final melt-pool pair, (i.e., 25 out of 26 pairs at maximum), the first melt-pool of the visible group is the reference point for approximate surface positions relative to the midpoint of the turn.

3.1.6 Scan Process

The interior of the chamber, inlet optic, and plate surface were cleaned with isopropyl alcohol to minimize contamination by dust/other particulates. The build plate was secured to the platform and leveled while the scan plates were set on the chamber surface. The build plate was marked with a sharpie for scanning the outline of 1x1 plate as a centering aid. After scanning the centering aid and between each plate scan, gas was shut off to reduce excess pressure so the system's glovebox could be unlocked, and scan plates could be repositioned. Water coolant was enabled for lasers and optics and the system was purged using UHP Argon. All scans were run after oxygen content decreased below 1000ppm, the chamber achieved 50mbar positive pressure, the argon flow rate was 5L/min, and the flow speed was 1.3m/s. The design of build files is discussed in greater detail in section 3.3, but part parameters were imported and overwritten by toggling the CLI+ option in Aconity Studio (machine software).

3.2 Offline Approach for Ramp Definition

Much of the referenced work for this study showcased advances in control using custom hardware and software enabling real-time control of laser power, online path-planning, etc. However, current commercial systems still lack advanced features and transparent control for operators and researchers. The focus then of this study was adaptation of methods explored using

research systems, onto a commercial system which constrained the avenues of approach. A methodology was laid out taking advantage of the system's integrated monitoring capabilities. Given 2 key process controls, laser power and scan speed, a target of achieving improved consistency in melt-pool geometry was chosen. Using the relation of power to velocity based on an assumption of constant normalized enthalpy, a comparison between a ramped scan, ideal processing, and an upper bound, entering keyhole regime, or lower bound, entering lack of fusion regime, could be realized. The proposed workflow was to process acquired monitoring data from the scanner, approximate the unidirectional velocity profile when accelerating, and use the relation of power to velocity with assumed constant normalized enthalpy to develop ramps.

3.2.1 PCD Processing

The SP-ICE-3 controller on the MIDI+ uses the SL2-100 protocol to communicate to system hardware. A non-proprietary protocol, XY2-100, offers similar characteristics, transmission frequency of 100kHz and a 2 Mbps data rate, but lower positioning resolution, 16-bit vs 20-bit. All monitoring data is stored using Aconity's point cloud data (PCD) file format. The data used in this study was the x, y, and z position data from the 3D scanner's encoders, the laser state (S), and the laser power (P). The next figure depicts the data processing required for developing a simple power ramp.

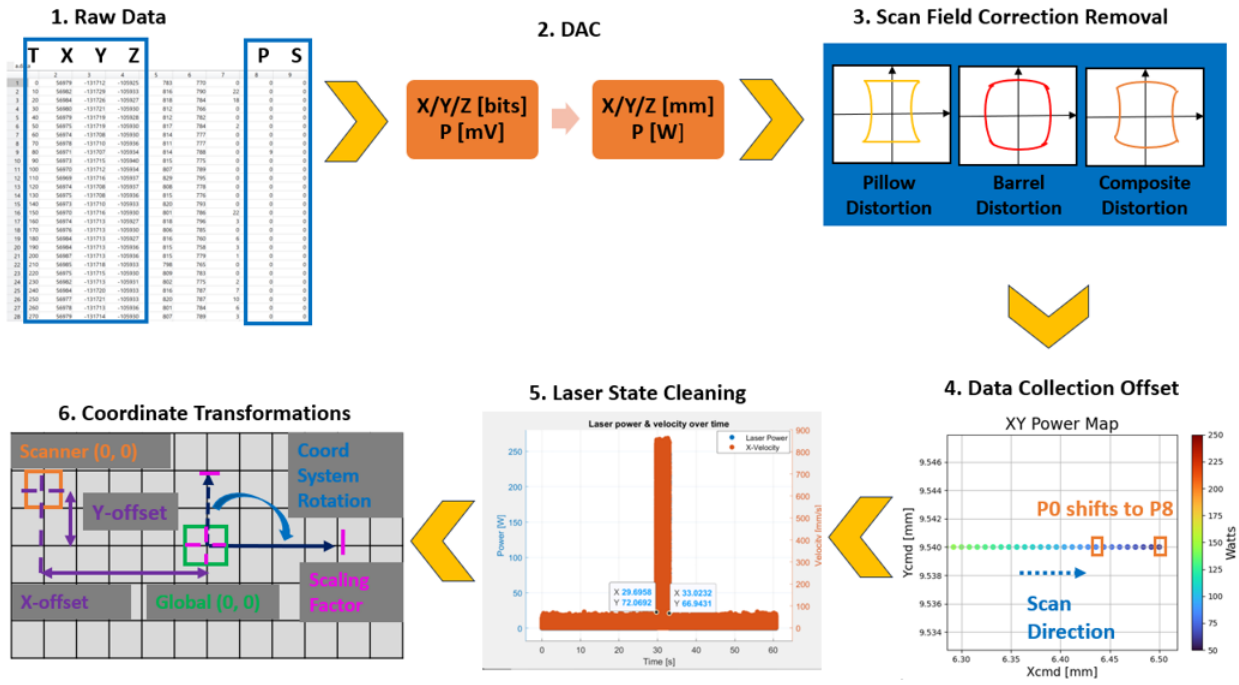

Figure 6: Monitoring data process map.

Figure 6 depicts each step of the process beginning with the raw position data which was recorded digitally (in bits), the laser state is binary, and the laser power is recorded in mV. The power conversion is more straightforward than for position since commands for the scanner compensate for optical distortions (applied field correction). However, it was determined that the collection of laser data occurs out of sync, by 80us, with scanner data after identifying inaccurate laser state with respect to position and verifying with timeseries data and skywriting plate scans which exhibited optimal laser delays. Details are provided in section 4.2.4, but the misalignment issue was not corrected until after the 2 exploratory scan sets discussed chapter 4. Figures using laser data produced from scans have been compensated post hoc to show the inaccuracy, but the commanded information used in the tests was derived from said inaccuracies.

Optical distortions originating from the arrangement of the motorized deflection mirrors in the scanner (galvos) and the focusing objective are represented as a composite distortion. The

distortion, if uncompensated, affects the accuracy of scanning compared to commanded geometry in the working field. Correction files are created to ensure the galvos move to coordinates which result in accurate positioning of the laser spot on the build platform. The process of retrieving the intended position data (i.e., the coordinates the galvos would have moved to if no distortion correction were required) is referred to as scan field correction removal. Aconity3D provides a python package with functionality for performing the removal. Additional transformations are due to the configuration of the Aconity system. Equipped with 2 scanners, each has a local coordinate system with a different origin. Application of an x and y offset, a scaling factor, and 90-degree rotation are required to represent the position data in the global coordinate system. MATLAB was used to perform the coordinate system transformations.

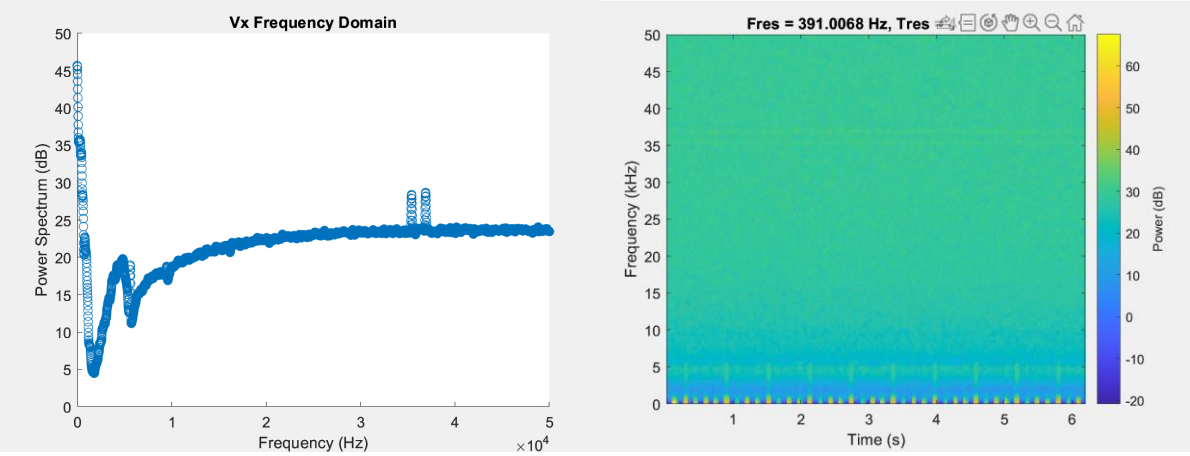

Figure 7: Power spectrum analysis for velocity data.

Calculating velocity data was performed with finite backward differences over the array of position data. This provides an approximation of the derivative of the position function, describing the scanner's motion, with respect to time at a temporal resolution of 10us. Although the position data was visually distinguishable, repeated use of numerical differentiation resulted in increased residual noise by multiple orders of magnitude. The resulting velocity and

acceleration data contained enough noise to present difficulty in identifying regions of interest (ROI's). Use of the "pspectrum" function in MATLAB for producing power spectrum plots or spectrograms aided identification of the cutoff frequency as seen by Figure 7. A cutoff frequency of 61 Hz was applied for the x-component velocity data based on a 3dB drop in power from the initial peak. The filter enabled accurate identification of bounds surrounding points of acceleration to separate steady state and transient regions. Figure 8 depicts the rapid drop off in signal power and the more distinct peaks in the time domain.

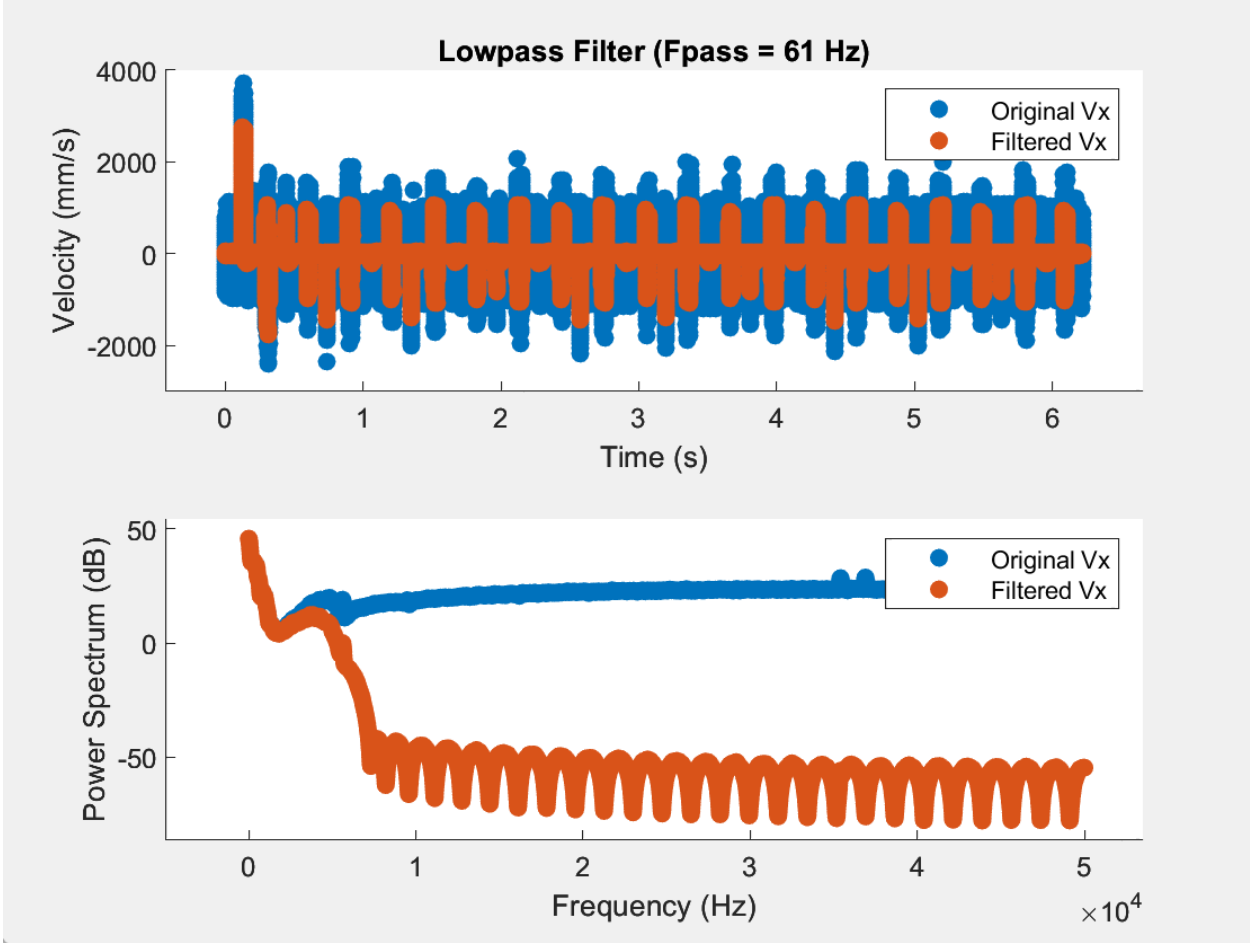

Figure 8: Comparison of raw vs filtered data.

3.2.2 Velocity Assessment

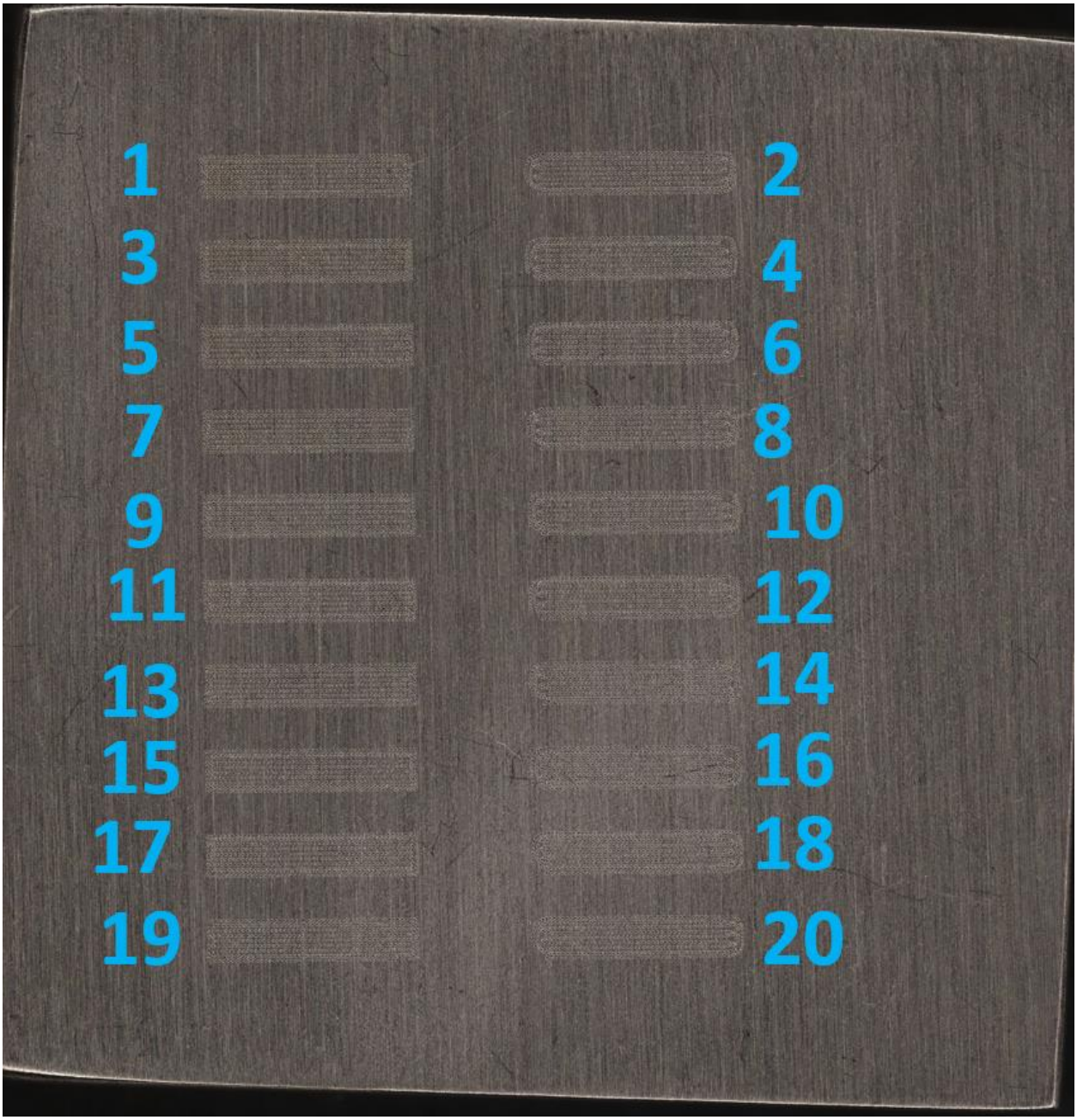

Figure 9: Pulsed hatch blocks with extension time = 500us (Left) and 0us (Right).

The approach used for implementing ramps on the MIDI+ considered modification of skywriting parameters rather than modifying scanner delays, to force the system to continue marking in acceleration regions. The skywriting extension time parameter determines time

provided for performing the movements performed in skywriting, acceleration events. When reduced to zero the skywriting motion is executed earlier, which results in marking during the motion since the difference in time moves the region spatially into the part of the vector where the laser is in an on state. This results in a constant build mode similar to what was described in [18]. At the cost of geometric accuracy, the marking of a region with nonconstant velocity provides an opportunity to vary power to achieve consistent melt-geometry and compare it to the steady state region.

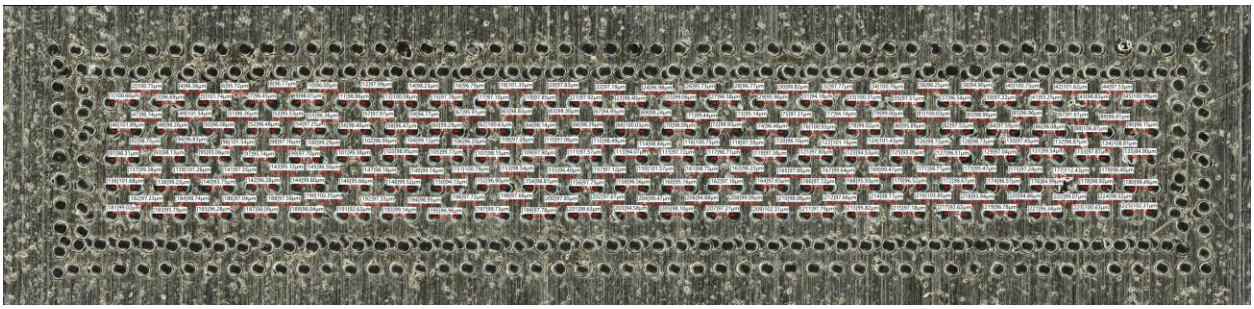

Figure 10: Pulsed scan with extension time = 500us.

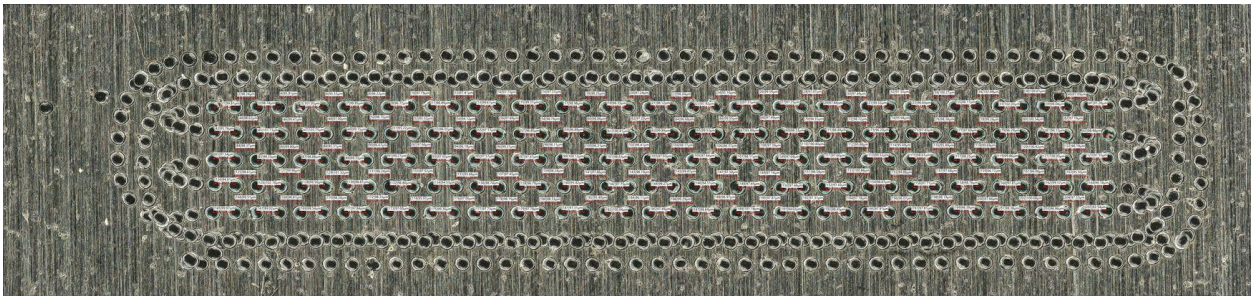

Figure 11: Pulsed scan with extension time = 0us.

Prior to the evaluation of the changing velocity profile, the accuracy of scanner data was verified experimentally. 20 hatch blocks were scanned, Figure 9, with constant laser power of 200W, scan speed of 1000mm/s, skywriting enabled, but an extension time of 500us (default) for half and zero for the other half. The laser was modulated at a frequency of 10 kHz and a pulse width of 10us, duty cycle = 10%. The length of the marks, dictated by the pulse width, was not relevant so long as there is a visible gap between them, enabling center-to-center measurements.

The resulting marks, Figures 10-11, should be spaced ~ 100um apart. Average measured distance was used to calculate velocity assuming laser modulation execution was exact, errors were based on nominal parameters, and comparisons between physical scan results were compared to 2 methods using the digital monitoring data. The steady state portions for 5 of the 7 hatch vectors, Figures 12-13, were used for analysis of regions 1 and 2.

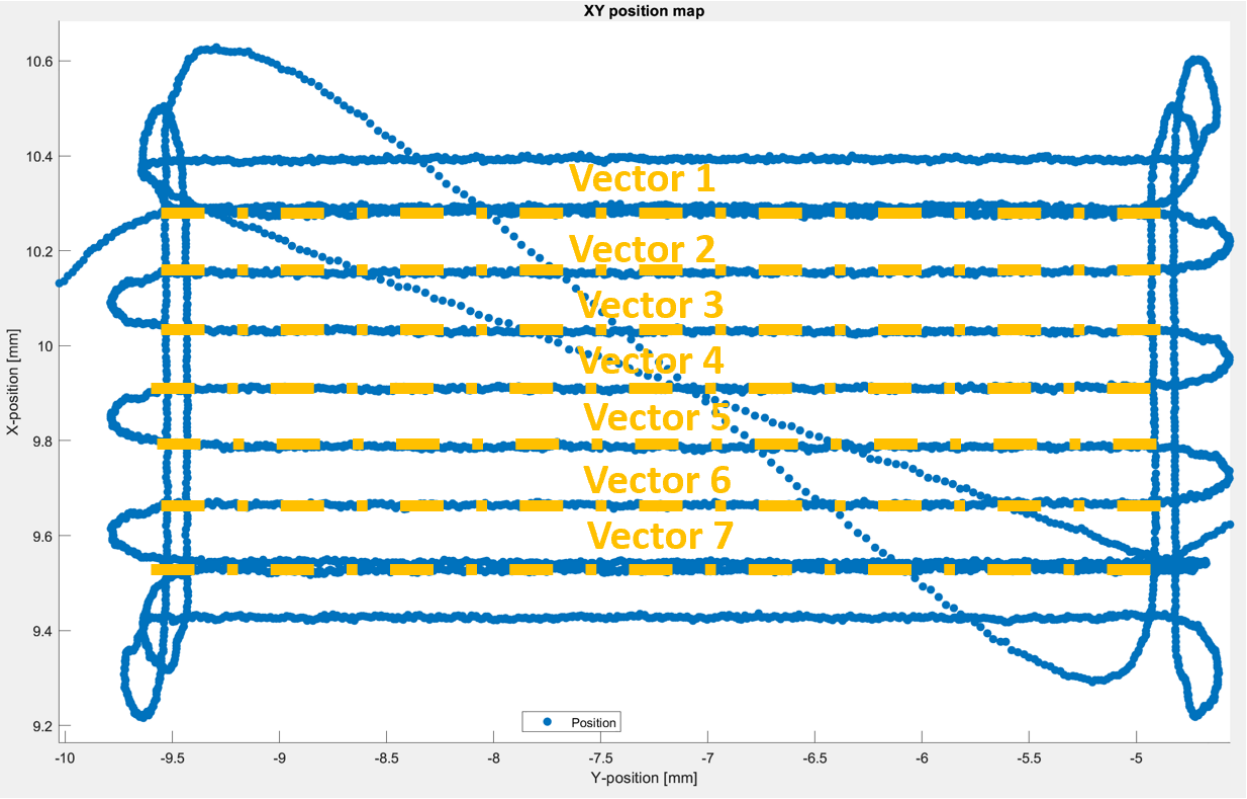

Figure 12: Position data for region 1 (note axes were flipped prior to rotation).

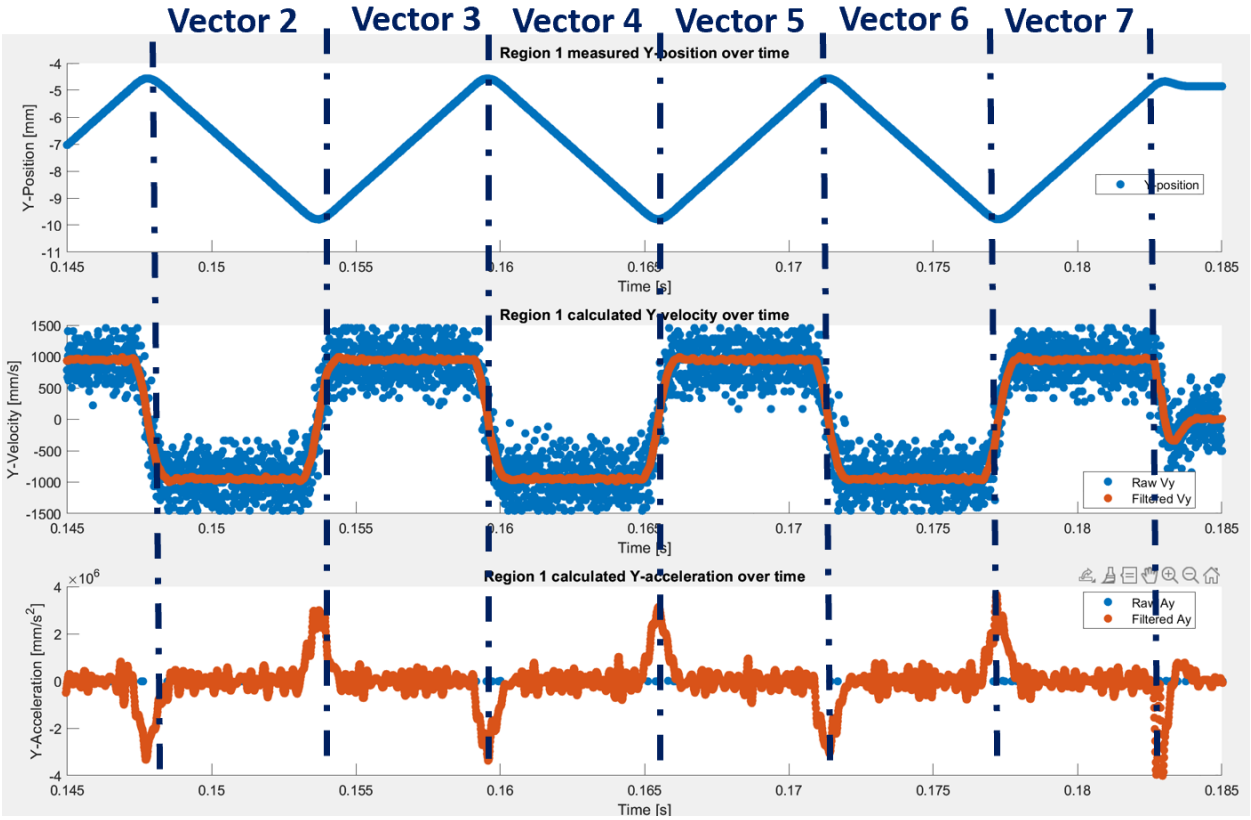

Figure 13: Window for steady state region (note axes were flipped prior to rotation).

Table 1: Measured velocity for 500us region.

Vector [#]	Max Velocity [mm/s]	Min Velocity [mm/s]	Mean Velocity [mm/s]	Standard Deviation [mm/s]
1	1017.4	932.5	979.1	20.2
2	1068.7	916.1	978.7	24.7
3	1088.1	849	977.2	31.6
4	1124.3	849.9	978.5	34.8
5	1038.5	926.2	977.6	23.3

Table 2: Error between commanded and measured velocity for 500us region.

Vector [#]	Max Error [%]	Min Error[%]	Mean Error [%]
1	1.74	-6.75	-2.1
2	6.87	-8.39	-2.1

3	8.81	-15.1	-2.2
4	12.43	-15	-2.1
5	3.85	-7.4	-2.2

Table 3: Measured velocity for 0us region.

Vector [#]	Max Velocity [mm/s]	Min Velocity [mm/s]	Mean Velocity [mm/s]	Standard Deviation [mm/s]
1	1024.9	912	979	20.2
2	1037.5	941.2	978	19.4
3	1004.2	915.5	977.8	17.6
4	1021	945.1	977.8	17.7
5	1011.8	925.4	977.9	16.1

Table 4: Error between commanded and measured velocity for 0us region.

Vector [#]	Max Error [%]	Min Error[%]	Mean Error [%]
1	2.5	-8.8	-2.1
2	3.8	-5.9	-2.2
3	0.4	-8.5	-2.2
4	2.1	-5.5	-2.1
5	1.2	-7.5	-2.2

The overall results, Tables 1-4, determined that average velocity calculated from measurements was 978.2 mm/s and 978.1 mm/s for the 500us and 0us regions respectively. This represented an ~ 2% reduction in velocity from the commanded value. Similar reductions were found for 2 methods used to calculate velocity from scanner and laser data.

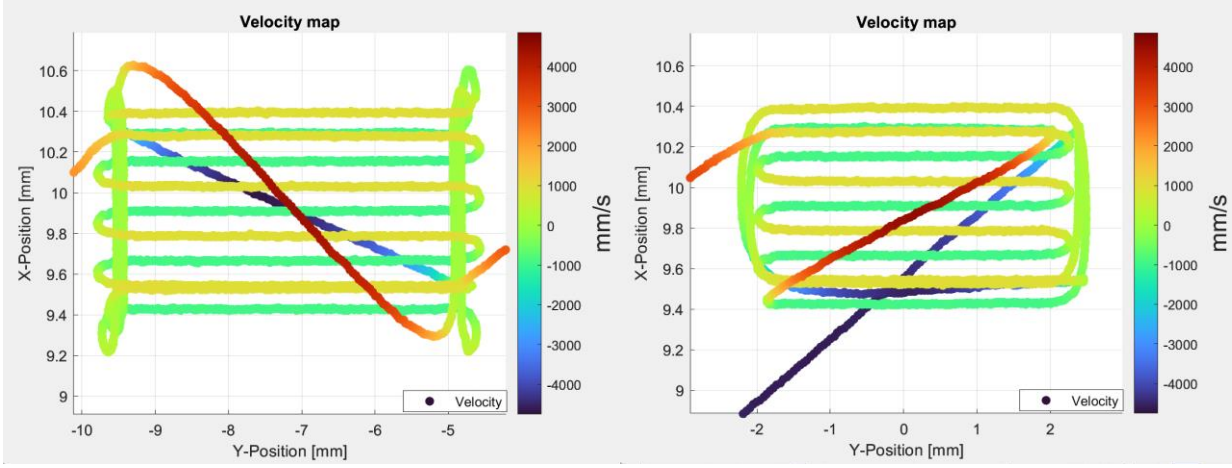

Figure 14: Velocity maps for the 500us region (left) and the 0us region (right).

The first method used the calculated velocity data after filtering, results shown in Figure 14 and Tables 5-8. ROI were identified by the start of acceleration or deceleration, 2 of the 7 hatch vectors which had been excluded in the physical results due to overlaps were similarly excluded in these results due to entering or exiting jumps at ends of vectors. The resulting 5 had bounds set and metrics for the data within the established windows were tabulated below.

Table 5: Method 1 calculated velocity for 500us region.

Vector [#]	Max Velocity [mm/s]	Min Velocity [mm/s]	Mean Velocity [mm/s]	Standard Deviation [mm/s]
1	998.1	911.1	951.7	15.5
2	995.1	915.5	951	15.6
3	993.2	908.9	950.7	14.7
4	991.3	913.5	950.8	15
5	996.1	912.6	951.3	15.3

Table 6: Error between commanded and method 1 calculated velocity for 500us region.

Vector [#]	Max Error [%]	Min Error[%]	Mean Error [%]
1	-8.9	-0.2	-4.8

2	-8.4	-0.5	-4.9
3	-9.1	-0.7	-4.9
4	-8.7	-0.9	-4.9
5	-8.7	-0.4	-4.9

Table 7: Method 1 calculated velocity for 0us region.

Vector [#]	Max Velocity [mm/s]	Min Velocity [mm/s]	Mean Velocity [mm/s]	Standard Deviation [mm/s]
1	990.9	914.6	948.9	16.7
2	988.4	917.5	948.7	15.3
3	990.6	909.8	949	12.7
4	987.5	906.9	949.1	14.9
5	995.9	918.6	948.6	13.3

Table 8: Error between commanded and method 1 calculated velocity for 0us region.

Vector [#]	Max Error [%]	Min Error[%]	Mean Error [%]
1	-0.9	-8.5	-5.1
2	-1.2	-8.2	-5.1
3	-0.9	-9.0	-5.1
4	-1.3	-9.3	-5.1
5	-0.4	-8.1	-5.1

The overall average velocity calculated from the filtered data was 951.1 mm/s and 948.9 mm/s for the 500us and 0us regions, respectively. This represents an ~ 5% reduction in velocity from the commanded value. The second method used the laser state signal, results shown in Figure 15 and Table 9. Identifying ends of vectors using a laser state position map and timeseries

data enabled calculation of the velocity for the 500us region, the 0us region was deemed too subjective due to the behavior at ends of vectors.

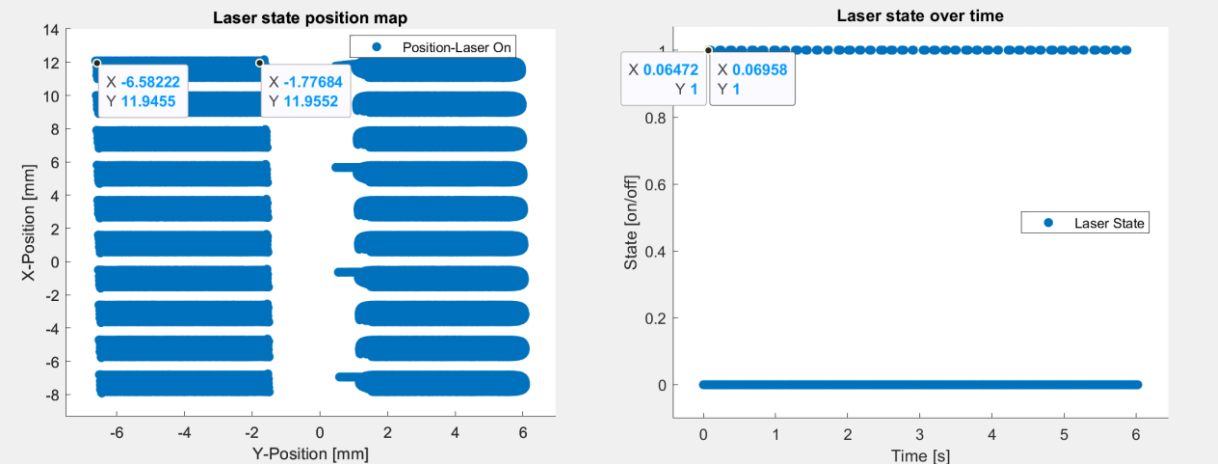

Figure 15: Laser state position map(left) and state timeseries data (right) for 500us region.

Table 9: Method 2 calculated velocity for 500us region.

Vector [#]	Laser state time delta [ms]	Laser state position delta [mm/s]	Mean velocity [mm/s]	Mean error [%]
1	4.86	948.9	988.8	-1.1
2	4.86	948.7	989.1	-1.1
3	4.86	949	988.1	-1.2
4	4.86	949.1	990	-1
5	4.86	948.6	989.2	-1

The overall average velocity calculated from the filtered data was 989 mm/s for the 500us region. This represents an ~ 1% reduction in velocity from the commanded value. Based on the spread of results, and the discrepancy between methods utilizing scanner position data, the -2% error from physical measurements was selected to represent expected error for future measurements. However, due to increased resolution, the process used for method 1 would be

extended for developing the velocity approximation and future accuracy assessment. In summary there is measurable variability in the scanner's velocity when operating in steady state regions. The -2% reduction, set as the expected scanner velocity tolerance. identified can be compensated for by analyzing in-process monitoring data. However, the other metrics presented indicate residual noise in velocity data even after filtering. This may complicate analysis of regions outside of steady state and impact the accuracy of approximated profiles as seen in the following section.

3.2.3 Velocity Approximation

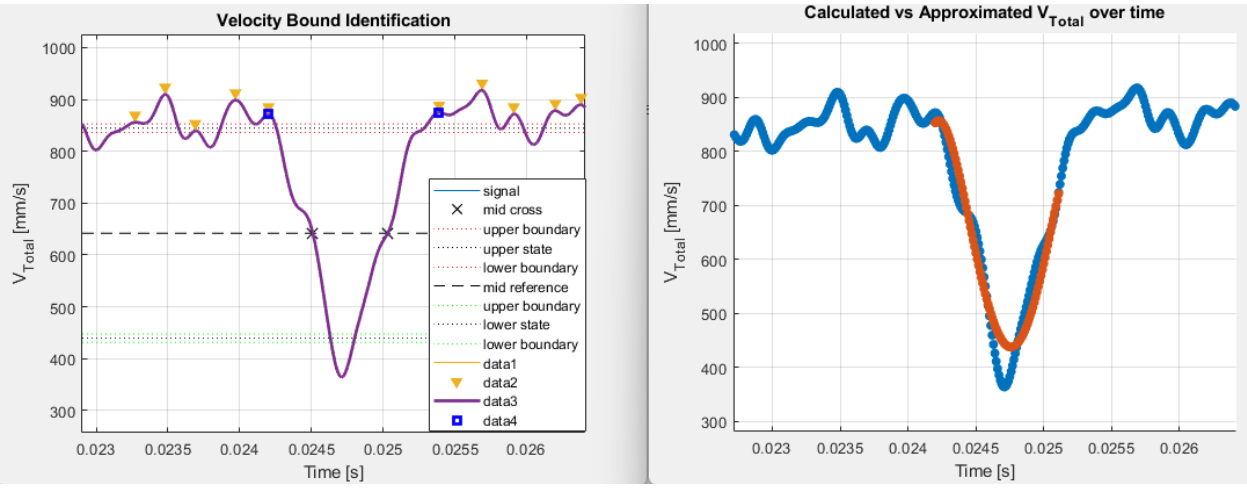

Figure 16: Polynomial approximation of deceleration profiles.

For broad applicability and flexibility, the approximation used must consider how the scanner enters and exits turnarounds for varying parameters. The general approach taken was scanning the test geometry on a plate to determine accuracy of baseline parameters, section 3.2.2, followed by analysis of larger data sets to define the approximation. For the experiments performed in chapter 4, the x-component velocity was used for analysis and approximation. Future experiments will consider total velocity for the combined x- and y-direction move in type 1 geometry, but the profile shown in Figure 16 is typical for those used in exploratory work. The

initial methods used to identify and quantify the profile were extremely simplified. Both the deceleration and acceleration portions of the profile were used as a single profile. Since only the x-component of velocity was used, if the y-component during processing was significant, the total velocity would have been higher, resulting in reduced energy density for the profile that was used as seen in section 4.1.4. Bounds for the profile were manually identified and a single profile was approximated and compared to data from other vectors. Figure 17 depicts an example of a linear approximation made for the first vector and compared across several others.

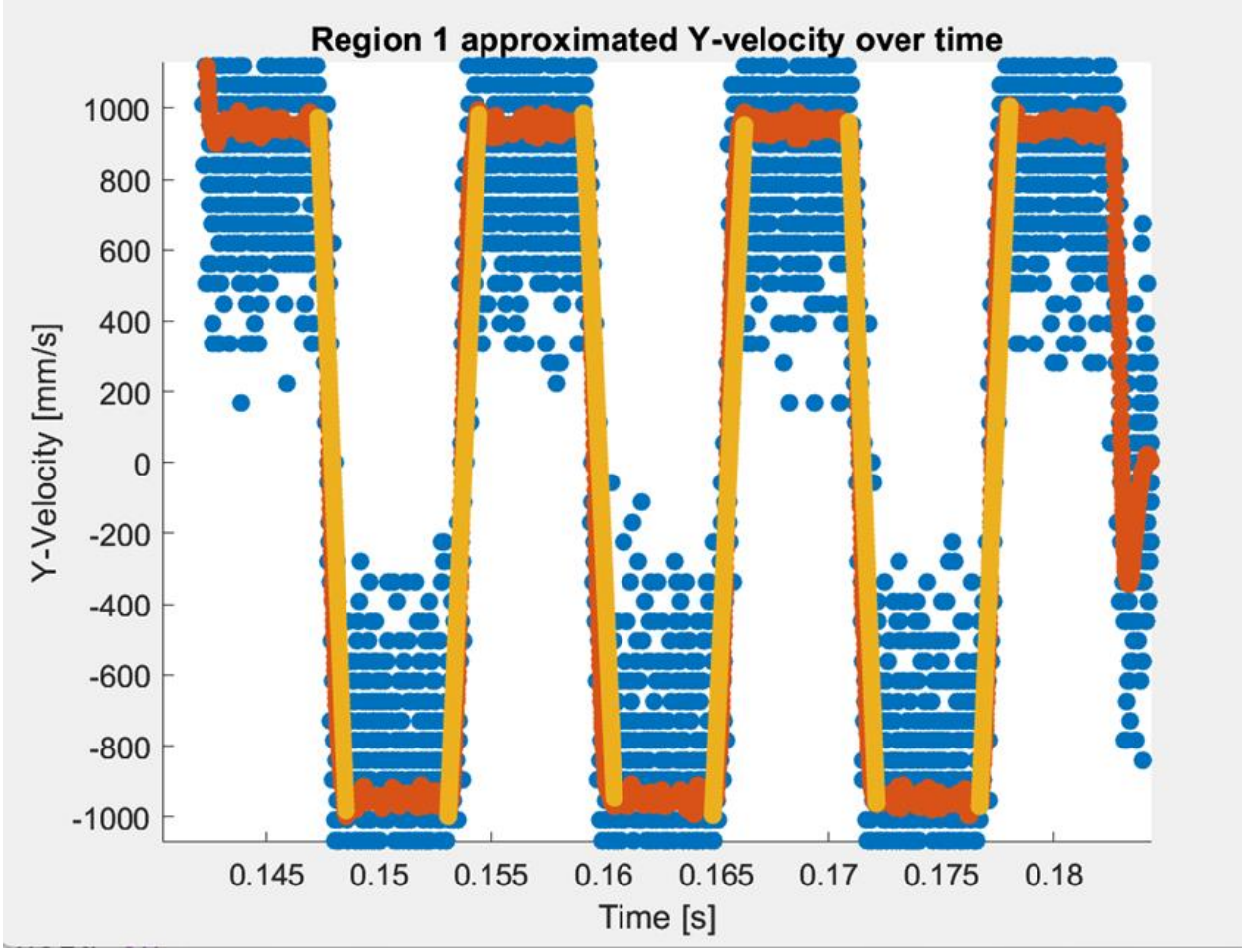

Figure 17: Linear approximation (note data axes were flipped prior to rotation).

Despite the limitations, it was decided that a rapidly produced test may support development of a more extensive kinematic characterization. Bounds at the ends of vectors were used to separate the steady state data from the velocity profile while accelerating. Filtered timeseries data within the established window was used as the argument for the "polyfit" function. This produces a polynomial approximation of n degree based on the following parameters p(x, y, n), where for this case x are query points (length of the profile), y are velocity values in the window, and a 4^{th} degree polynomial was selected. The function returns values for [P, S, Mu] where P are the coefficients of the polynomial, S can be used for error estimates, and Mu contains 2 values to center and scale query points which are input to the polynomial. They are used in conjunction with the "polyval" function which evaluates query points, x, using the polynomial, y, in p(x, y) and the values were stored in a variable "polyfit". Next the coefficient of determination was calculated for multiple vectors to validate the polynomial regression. Residuals were determined by the difference of each vector's velocity values and those of polyfit. The sum of the squared residuals and the total sum of squares (total number of values in vector's velocity array minus 1 times the variance of the array) were computed. Lastly R^2 = 1 – (SSresid / SStotal), the majority of values were > 0.9, however due to slight variation in profiles between repeated scans or even vectors within a scan, outliers such as vector 4 in region 2 may present themselves. R^2 values for each vector of each region as seen in Figure 18 were listed in Table 10. Adjustment of bound setting, approximation parameters, and averaging of profiles used for the approximation rather than for a single profile may reduce this variability.

Table 10: Goodness-of-Fit for velocity model of transient events in regions 1 & 2.

Vector [#]	Region 1	Region 2
1	0.9993	0.9948
2	0.9248	0.9975
3	0.9979	0.9198
4	0.9015	0.6956
5	0.9955	0.9969
6	0.9813	0.9929

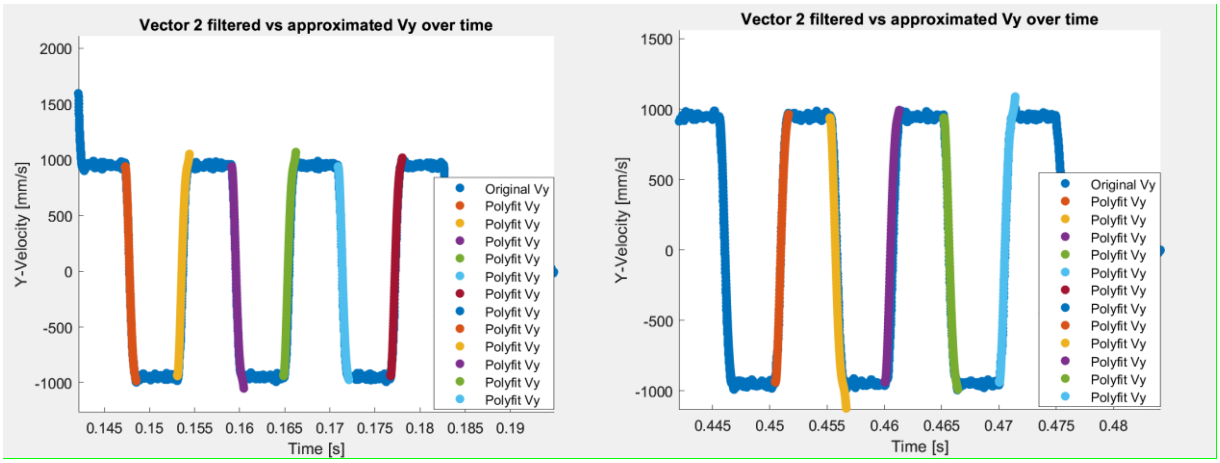

Figure 18: Velocity approximation for vectors of both regions.

3.2.4 Simplified Power Ramp

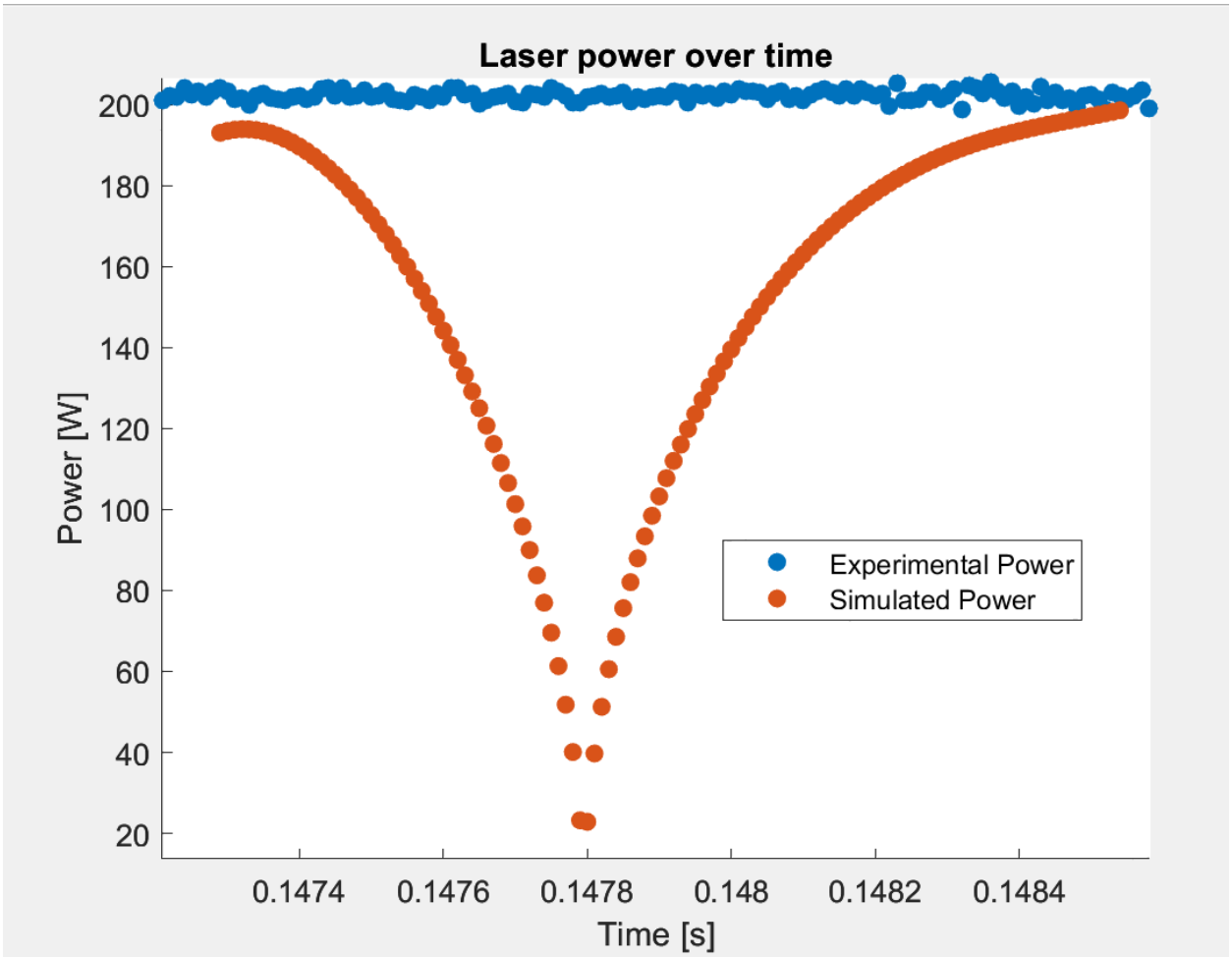

Figure 19: Power profile during acceleration events.

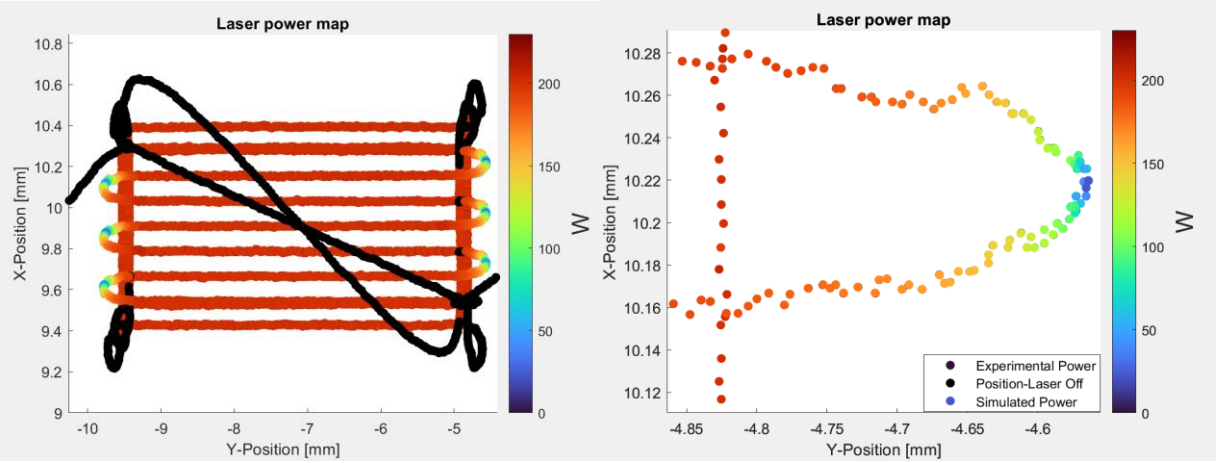

Figure 20: Simulated power ramp superimposed on CP data.

Recalling the constraints set in section 3.2, the simplified power equation is defined as the following: Psim1 = Pcmd * (sqrt(abs(Polyfit / Vcmd)));

where Psim1 represents adjusted power and is equal to the product of the nominal (commanded) power and the square root of the ratio of the approximated (considered actual) velocity to the nominal velocity. Note the classification of this as simulated power is best seen in Figures 19-20. Laser power reduces from the nominal value to a minimum of ~ 20W, but this does not account for a key hardware constraint which is the lower laser cutoff limit. When testing 60W was used to avoid potential issues when executing, which may result in excessive energy density deposited at the ends of vectors, although the time spent above the ideal powers in this profile is < 100us. As previously discussed, other concerns are the variability of velocity between vectors in the same scan. As such an upper bound was set to ensure any calculated powers greater than the nominal power were set equal to it. The use of only a single profile may have also introduced other complications which for the tests conducted in this study, did not manifest. The length of the windows set for various vectors between points of acceleration vary. By assuming all vectors were equivalent, application of the same ramp in different turns may result in turns where steady state velocity has been reached and power is still ramping or vice versa. For the purposes of exploratory experimentation, the potential difference in applied energy densities for those events was considered to occur at a negligible timescale compared to the length of the ramp. Table 11 below provides differences in profile length, although the profile of ramp 1 was used for all turnarounds during testing.

Table 11: Different profile lengths for vectors in same scan.

Vector [#]	Length of velocity profile during acceleration window [us]
1	1260

2	1380
3	1380
4	1400
5	1230
6	1330

3.3 Machine Implementation

All the work up to this point would be largely irrelevant if a system with real-time control/monitoring capabilities was in use. However, the selected commercial system provides functionality which can take advantage of the developed profiles. The only remaining challenge was modifying the simulated power equation with hardware constraints in mind and determining how to provide build data to the system. Aconity3D uses the common layer interface (CLI) file format to send command data to the system's controller. The coordinates constituting the geometry data is communicated using the SL2-100 protocol to the scanner, and the power data is sent to the laser. It was determined that the work required to produce the simplified test geometry described in section 3.1.5 would be reduced if both the coordinates and powers were produced in a simple program, rather than modifying files containing just geometry information from commercial software.

3.3.1 CLI Format & CLI+ Extension

The build file format used by Aconity systems are .ilt which is a container format that can be renamed to .zip to extract 2 types of files typically produced through software such as Autodesk Netfabb. The first file is a text file containing nominal build parameters such as scan speed, laser power, defocus, etc. for as many parts and vector types as were included in the build. The second file is a .cli file which contains a 2.5D representation of 3D components. Modification of parameters in AM software, or the parameter.txt file, or at the machine in the

Aconity Studio software are applied on a part basis, not for vectors within a part's layer. This limitation is key to the goals of this study and foundational existing work. Inability to vary process parameters between vectors may result in suboptimal material processing or defect formation when multiple conditions that represent differences between those vectors are not accounted for. Specifics regarding the file format and Aconity 3D's implementation are available from [35] and [36] respectively. For the purposes of this study, the decomposition of vectors into polyline chains were used in combination with the "$$POWERS" extension to vary power between vectors at the maximum available timescale of 10us.

3.3.2 Scan Strategy Requirements

Elaborating on the use of the format in conjunction with the intended type 1 geometry, the purpose of decomposing vectors is based on the limitation of the specific extension used. "$$POWERS" applies a series of powers separated by indices to a set of coordinates representing vectors. Power at index 0 applies to vector at index 0, which would not adjust power between 2 vectors in a turnaround more than once. To increase the temporal resolution of power adjustment such that the system can execute the developed profile necessitated discretization of a pair of vectors (representing a turnaround between them) into a chain of polylines or micro vectors. In testing the maximum temporal resolution is 10us based on the communication protocol. This is not the most optimized method for the controller to process or for data storage since only short regions of vectors are the target for ramps and there exists potential for errors depending on how the geometry is decomposed. This will also affect the length of polylines depending on the nominal scan speed. Considering the parameters used in experimentation, at 1000mm/s 10um should be traversed over 10us. This was the difference used between vectors for these tests, but in reality, given the variability of the scanner in steady state and the changes during acceleration, the executed strategy will differ from the ideal. However, by increasing the

number of vectors, the number of potential power adjustments increases, which accomplished our goal of achieving a maximum control rate in the ROI.

3.3.3 Build File Generation

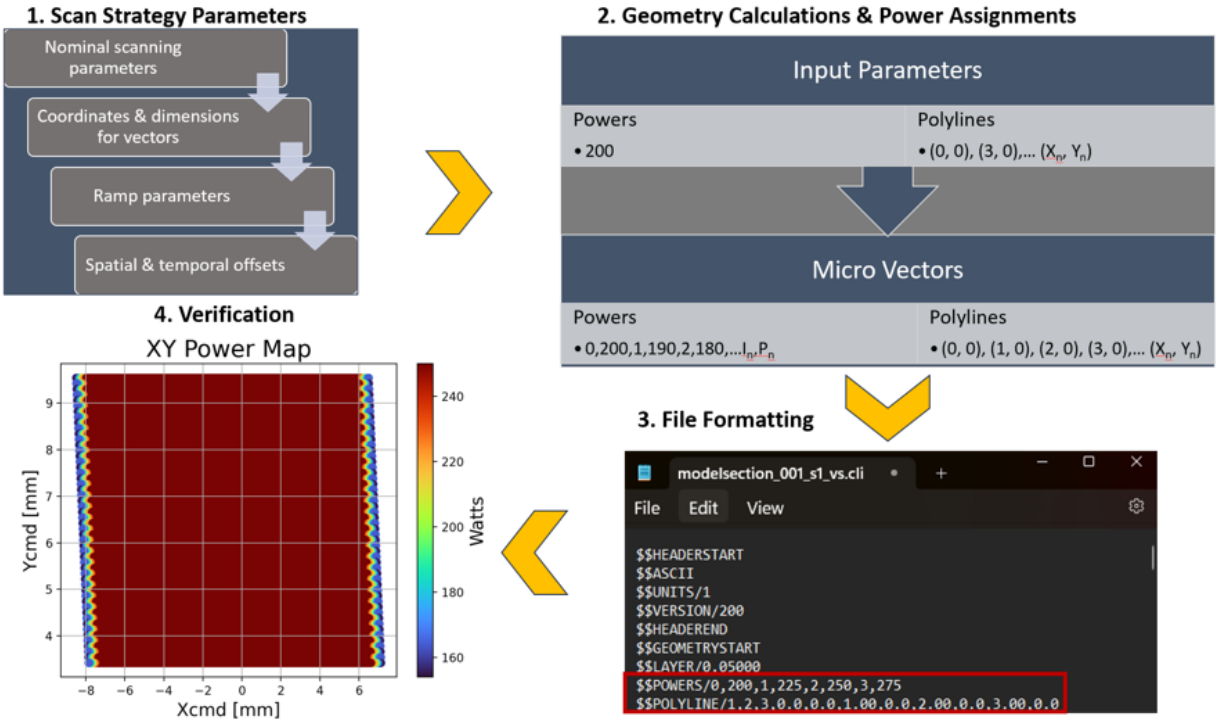

Figure 21: Build file generation process map.

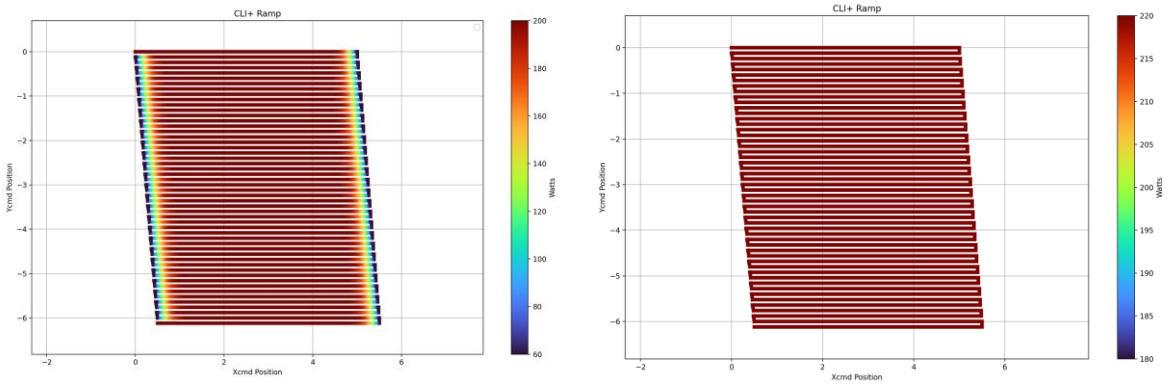

Figure 22: Build data for ramped (Left) and constant (Right) power.

The build file generator required the following variables for calculation of geometry and laser powers:

1. Initial position along x-axis (1st vector origin)
2. Initial position along y-axis (1st vector origin)
3. Length along x-axis (vector length since only unidirectional testing was performed)
4. Length along y-axis (hatch distance)
5. X-offset (for staggered section geometry)
6. Nominal scan speed (mm/s)
7. Nominal laser power (W)
8. Number of powers (timesteps)
9. Ramp offset (timesteps

Based on the 9 inputs above, the simplified geometry could be calculated and assigned powers as seen in Figure 21. The power equation was integrated along with hardware constraints to ensure powers remained within bounds. The resolution of control, or rate at which power was adjusted could be adjusted but for all testing performed in this study it was set to maximum. List comprehension and functionality of the "zip" class was used to organize the data in the appropriate format before writing to the file. Although data could be visualized immediately after calculation, the function was defined to parse the build file and verify data after it had been formatted. This served as an added check and to test functionality of interest discussed further in future work. Figure 22 depicts the 2 build files used for the first scan set detailed in section 4.1.

3.3.4 Proof-of-Concept

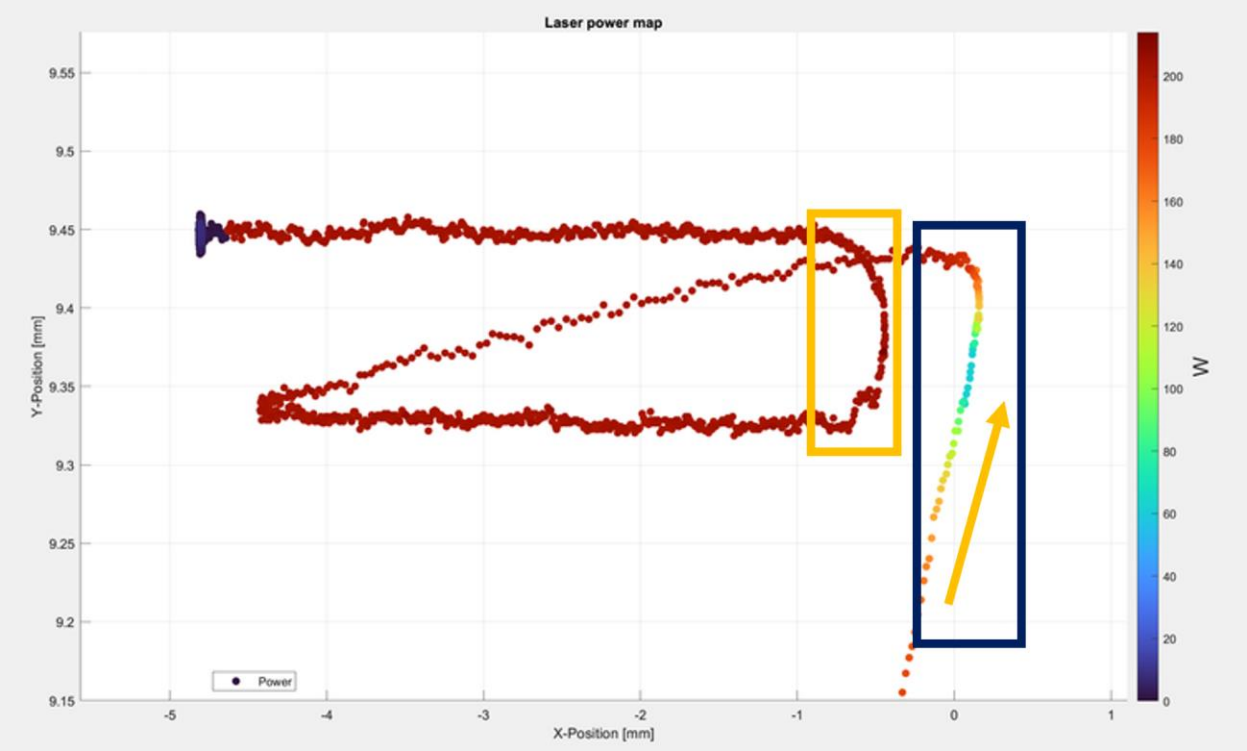

Figure 23: Power map of the first successful polyline-based ramp.

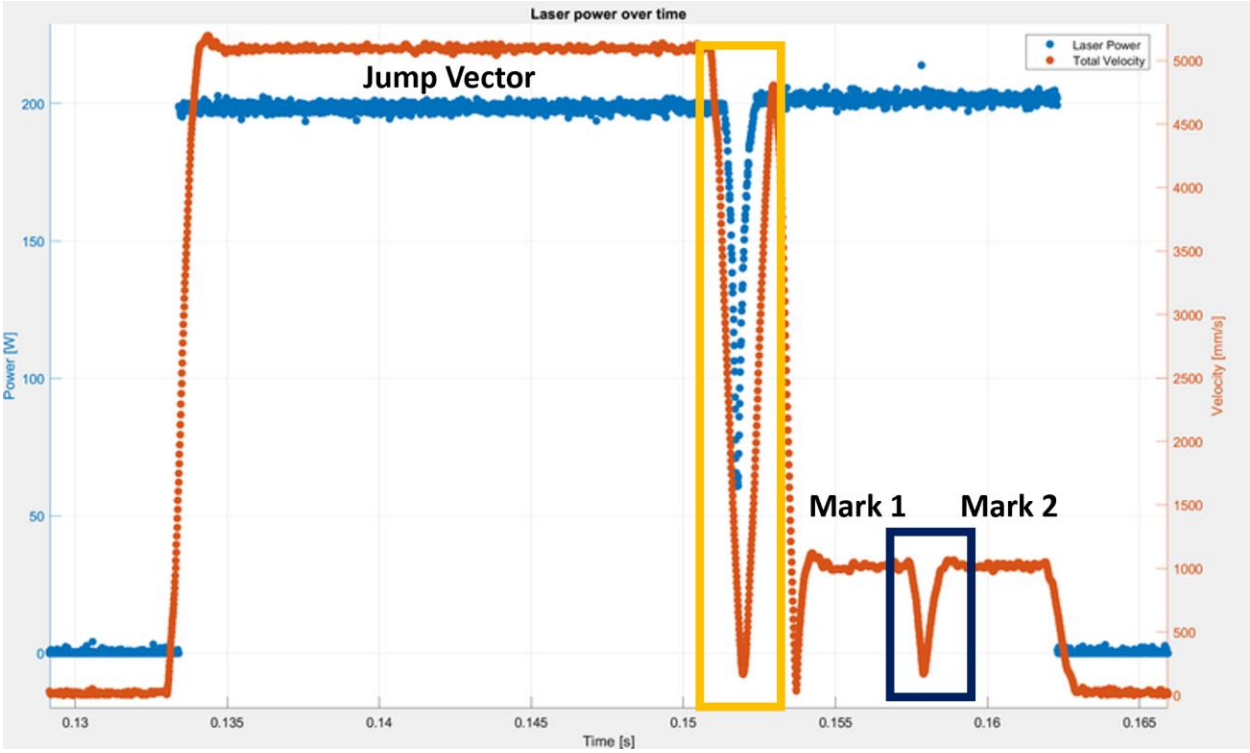

Figure 24: Misalignment between acceleration region and power ramp.

As a proof of concept, a 2-vector test was run to validate proper ramp execution. Unfortunately, the data showed misaligned scanner and laser ramps with respect to time as shown in Figure 23, where the scan direction followed the yellow arrow, the black box surrounds the position of the executed ramp, and the yellow box surrounds the intended location of the ramp. The cause of the delay, from this point forward referred to as ramp offset, was not identified, but a workaround was to shift the power indices in the build file such that the minimum power was aligned with the minimum velocity. In Figure 24, the decrease in laser power surrounded by the yellow box, must be aligned with the deceleration event surrounded by the black box between mark vectors 1 and 2. Repeated tests revealed that given the same geometry and ramp profile the adjustment is consistent between multiple trials, but can vary between vectors within the same scan, implying that the ramp profile issue regarding differing lengths of acceleration regions can affect accuracy.

CHAPTER 4: EXPLORATORY WORK & RESULTS

4.1 Validation Scan V1

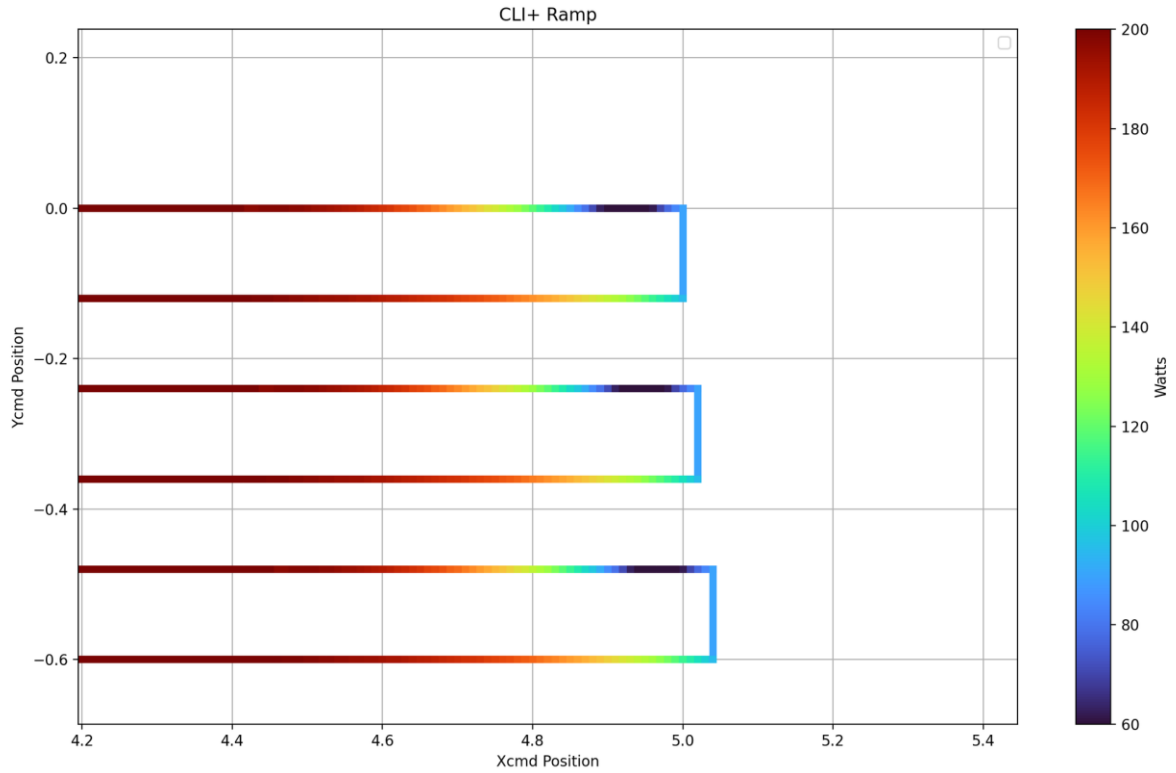

Figure 25: Ramp type 1 with ramp offset = 0us.

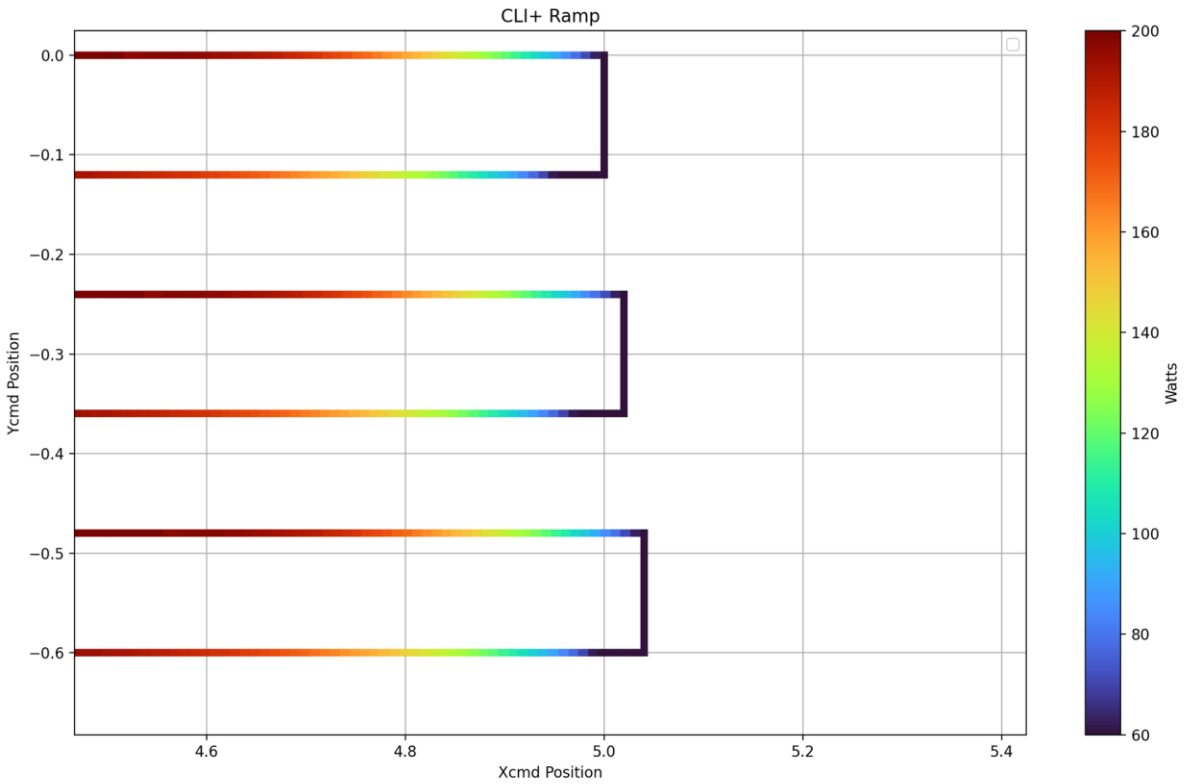

Figure 26: Ramp type 1 with ramp offset = 100us.

Validation scans were performed to determine the following:

1. Accuracy of power/velocity values during ramp.
2. Surface geometric accuracy.
3. Melt geometry variability in ROI.

A CP vs RP comparison test was run with 26 pairs of vectors, each offset by 20um for sectioning. The files had equivalent coordinates and the only testing difference was the application of RP at ends of vectors (in the turn) for the ramped file. The ramp offset had been applied, which shifted the minimum power over the hatch distance move as seen by Figures 25-26. The length of scans was shortened due to a reduction in extension time (i.e., the portion of the scan where skywriting normally occurs is shifted 500us back when the laser state is still high). Therefore, at the cost of the equivalent distance scanned in steady state for the reduction, the laser will remain on during a turn.

4.1.1 Monitoring Results

Results of the validation scan version one (VSV1) included surface and cross-section measurements in addition to PCD data analysis. Automatic identification of the ROI proved difficult due to the residual noise in the velocity & power data, so ROI were manually identified considering the closest inflection points to the mid-cross point of the data series. In the steady state regions, the power and velocity for the first 5 turnarounds were analyzed, Figures 29 & 31. The average power for the RP scan was ~ 200.3W and for the CP scan was ~ 201.6W. The average velocity for the RP scan was ~ 988.3mm/s and for the CP scan was ~ 992.8mm/s. The maximum error from the nominal parameters for the RP scan was ~ 0.2% for power and -1.2% for velocity. Error for the CP scan was 0.8% for power and -0.7% for velocity.

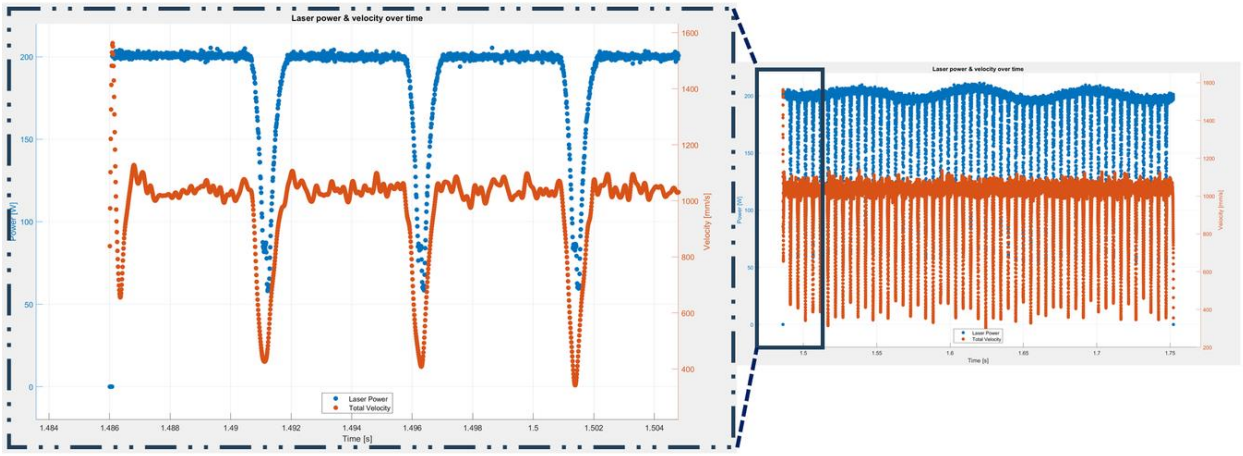

Figure 29: Power and velocity data for RP VSV1.

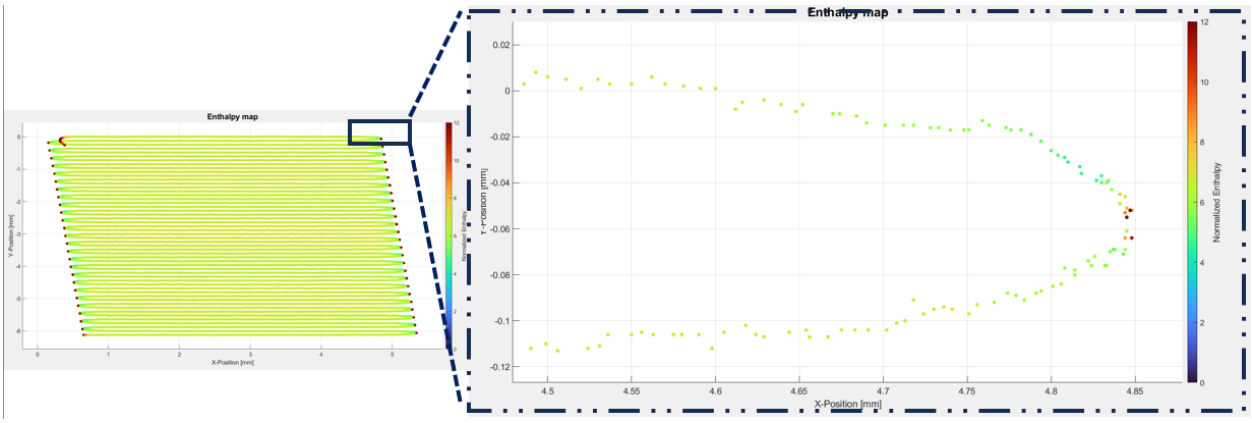

Figure 30: Calculated enthalpy for RP VSV1.

The power ramp is seen although due to steady state noise its length and start/stop time/position could not be confirmed. Starting at the nominal power setting of 200W it decreases to ~ 60W at the midpoint of the turn before increasing after the scanner changes direction. Rather than assuming an enthalpy target, all parameters of the model were held constant except for power and velocity. However, the intention of the power calculation was to keep that ratio, and by extension normalized enthalpy, constant. Unexpectedly, enthalpy decreased in all turns due to an error made in approximation, Figure 30, where enthalpy was calculated with respect to total velocity, the approximation was made based on x-component velocity. This resulted in a reduction of ~ 50% which, if the y-component of velocity during the turn is critical, may severely impact maximum depth results.

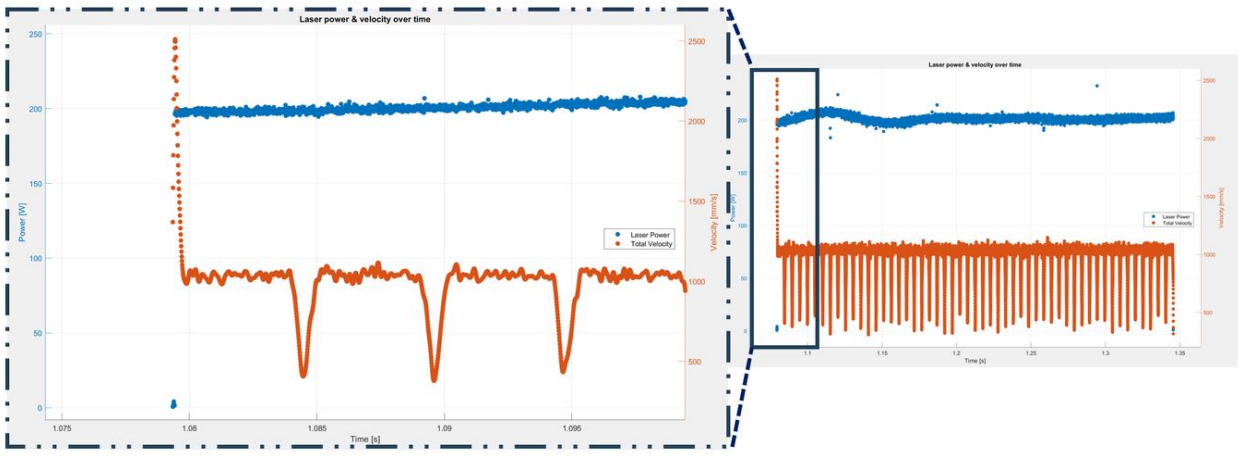

Figure 31: Power and velocity data for CP VSV1.

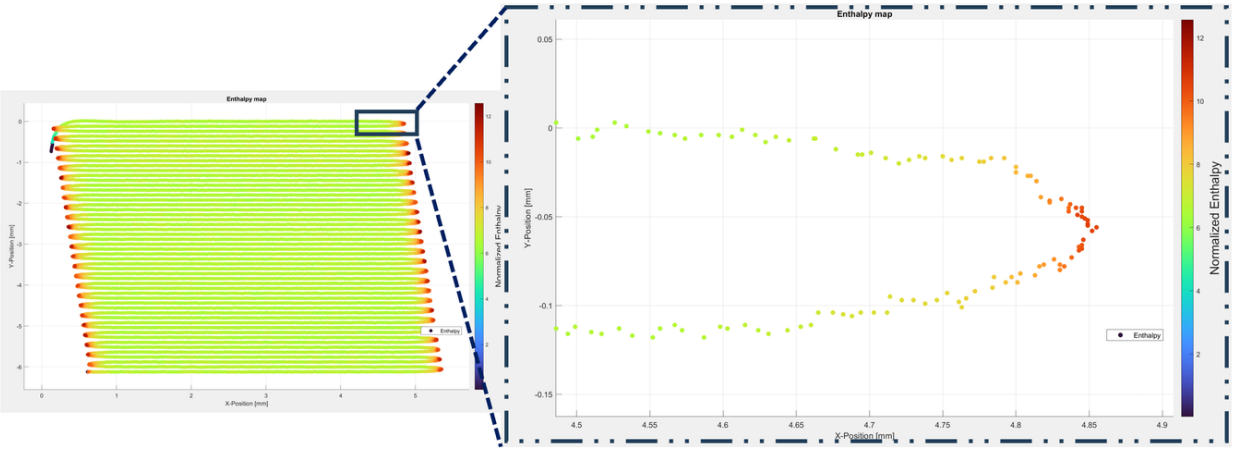

Figure 32: Calculated enthalpy for CP VSV1.

Compared to the steady state regions in the CP scan, calculated enthalpy increased by ~ 80% in the turn, from 6.5 to ~ 12 in Figure 32. Several studies using SS316 set bounds for entering the keyhole regime at enthalpy values of 16-20 and even as great as 30. Given the metallography results, it's unlikely that this increase was sufficient to form that defect, but the heat accumulated during the turn did have a measurable effect as seen in section 4.1.3.

4.1.2 Surface Results

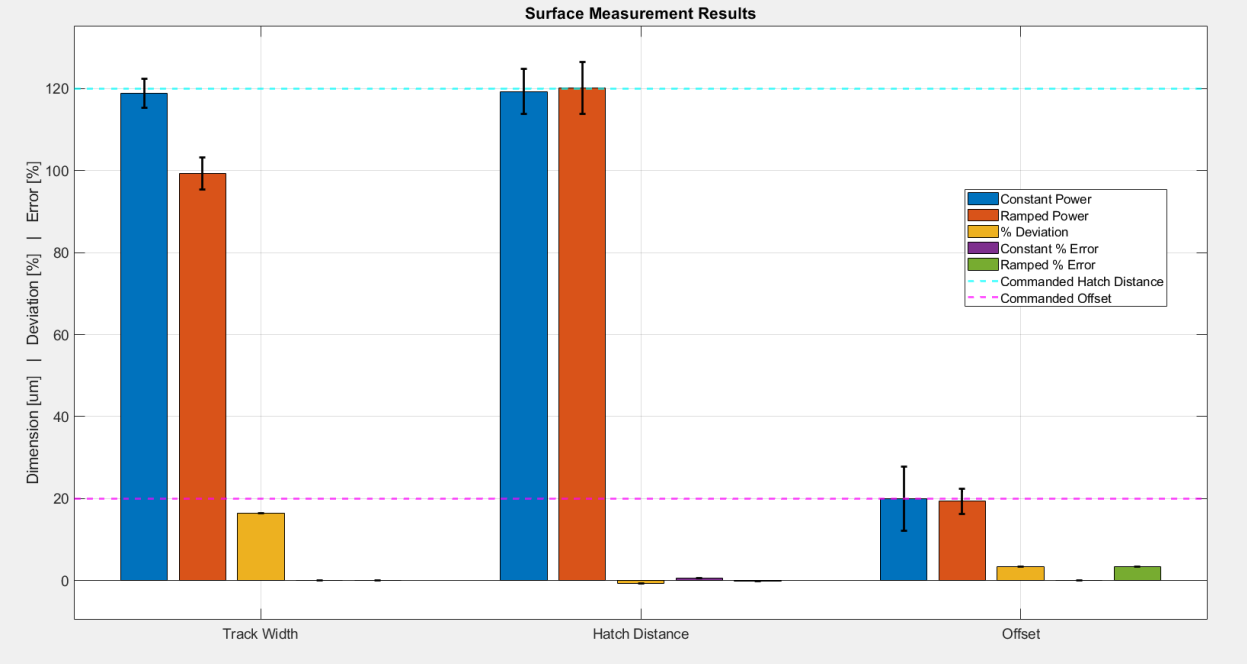

Figure 33: Surface measurement results for VSV1 scans.

Three dimensions were considered for surface analysis: vector offset, hatch distance, and track width along the center of the steady-state region. The first 10 vectors of each scan had measurements averaged in Figure 33. The 2 coordinate-based values, vector offset and hatch distance, had a maximum error from nominal of 3.3% and -0.6% respectively. A discrepancy in track width deviated 16.4% between the ramped and CP scans. This resulted in increased overlap for the CP scan. The z-position of the 3D scanner and coordinates from the build files were verified to have remained constant between tests. Considering the small change in power for the CP scan, and the fact it only occurs for the first part of the scan, the error was attributed to operator error.

4.1.3 Section Results

Sectioning and grinding from the plate edge to the dotted yellow line enabled viewing melt-pool geometry at multiple distances along the length of equivalent tracks. Based on the

target offset along the x-axis, sections provide ~ 20um resolution between each melt-pool image. Although the exact start position cannot be verified, so long as the total amount of visible melt-pools is less than the total number scanned, the starting location of the first section must be ~ 0 to 20um from the edge of the turnaround as seen by the section line relative to the edge of turnarounds of vector pairs in Figure 34.

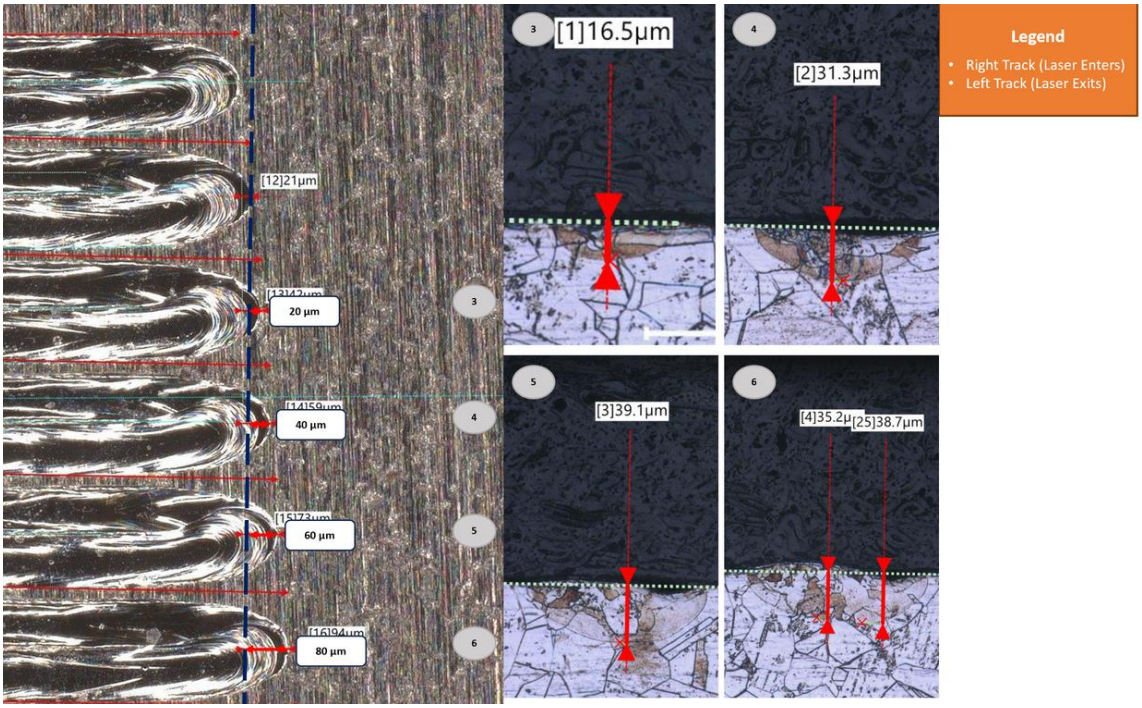

Figure 34: Melt-pools for tracks 3-6 for RP VSV1.

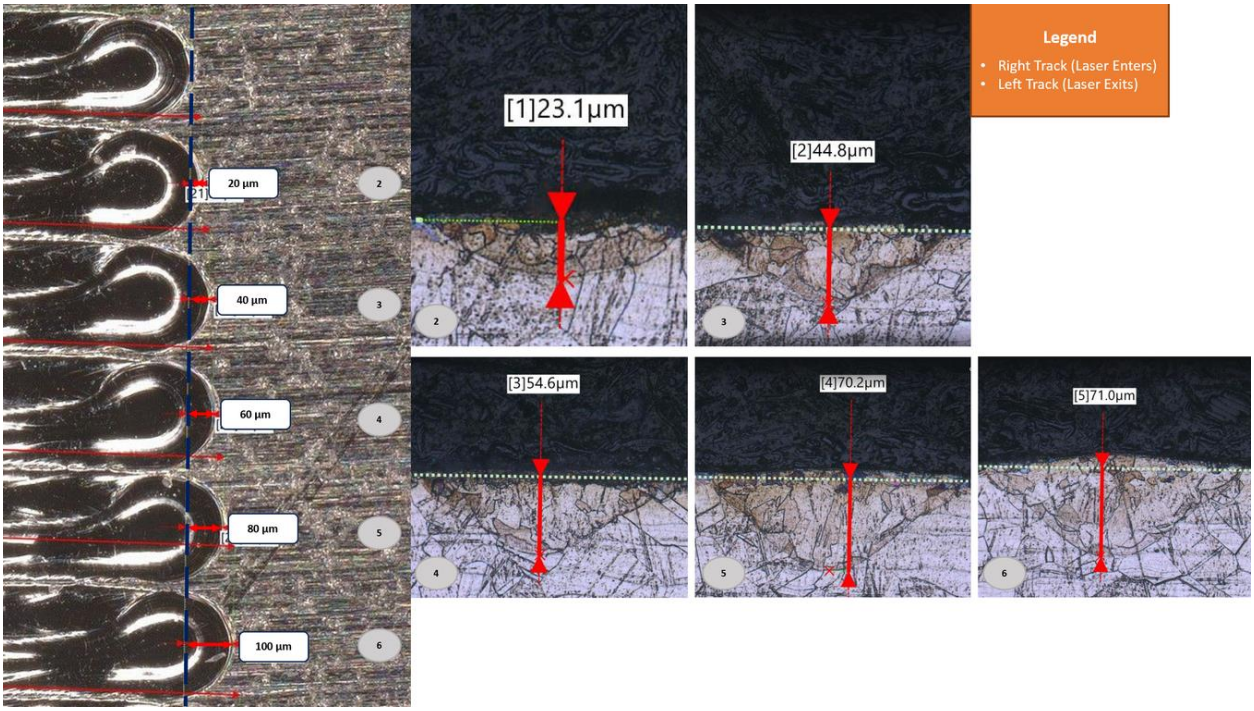

Figure 35: Melt-pools for tracks 2-6 for CP VSV1.

For both scans, ~ 60um into the track, only the edge of the melt-pool formation is visible. Additionally, at the midpoint of the turn the melt-pool is oriented parallel to the section rather than perpendicular. The first 3 images for each scan and measurement results were considered separate for calculated metrics. However, subsequent images in the CP scan, Figure 35, have single melt-pools in place of a pair, with a visibly large width to depth ratio indicating over melting. This large overlap continued until the 11th track for the CP scan, whereas the RP scan showed visually distinct melt-pools by the 4th track.

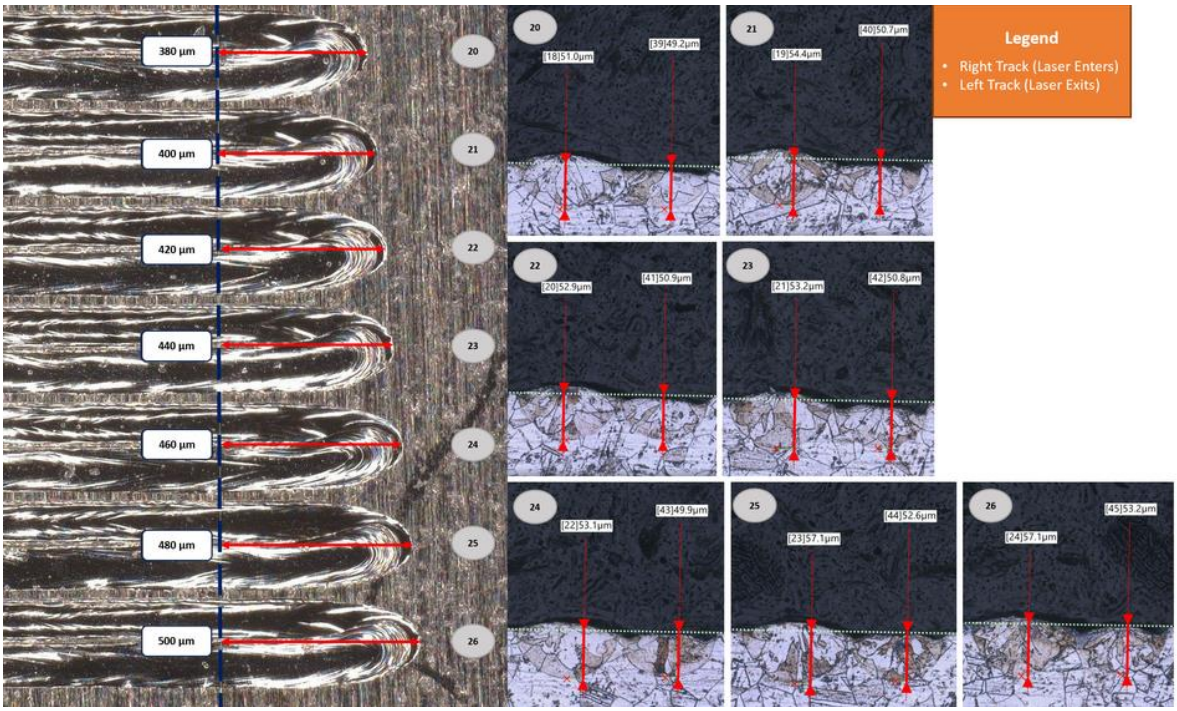

Figure 36: Melt-pools for tracks 20-26 for RP VSV1.

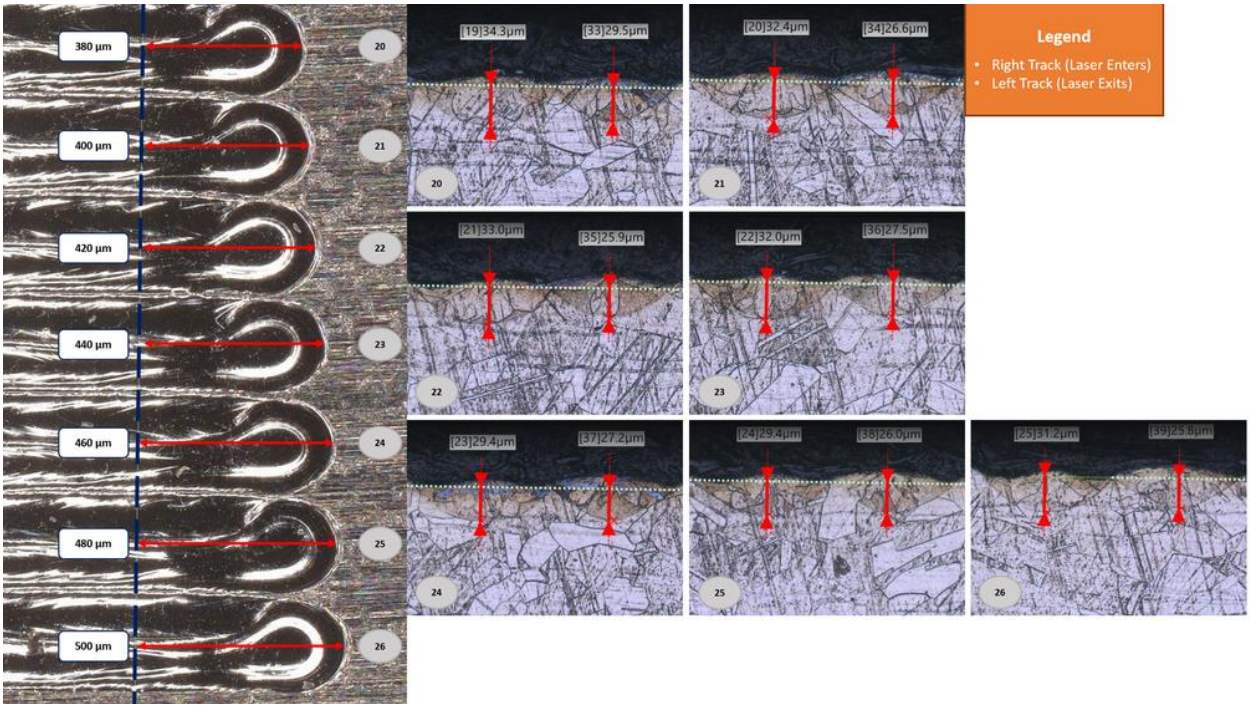

Figure 37: Melt-pools for tracks 20-26 for CP VSV1.

Considering the last 7 tracks from each scan, Figures 36-37, all tracks feature separate pairs of melt-pools, although the depth for the RP scan is higher on average. In addition to the discrepancy in track width, the reduced depth in the CP scan suggests the beam was defocused due to an error in build height. The position of the optical train with respect to the build plane was verified by checking the z-position of the 3D scanner, therefore the build platform itself was the most likely cause.

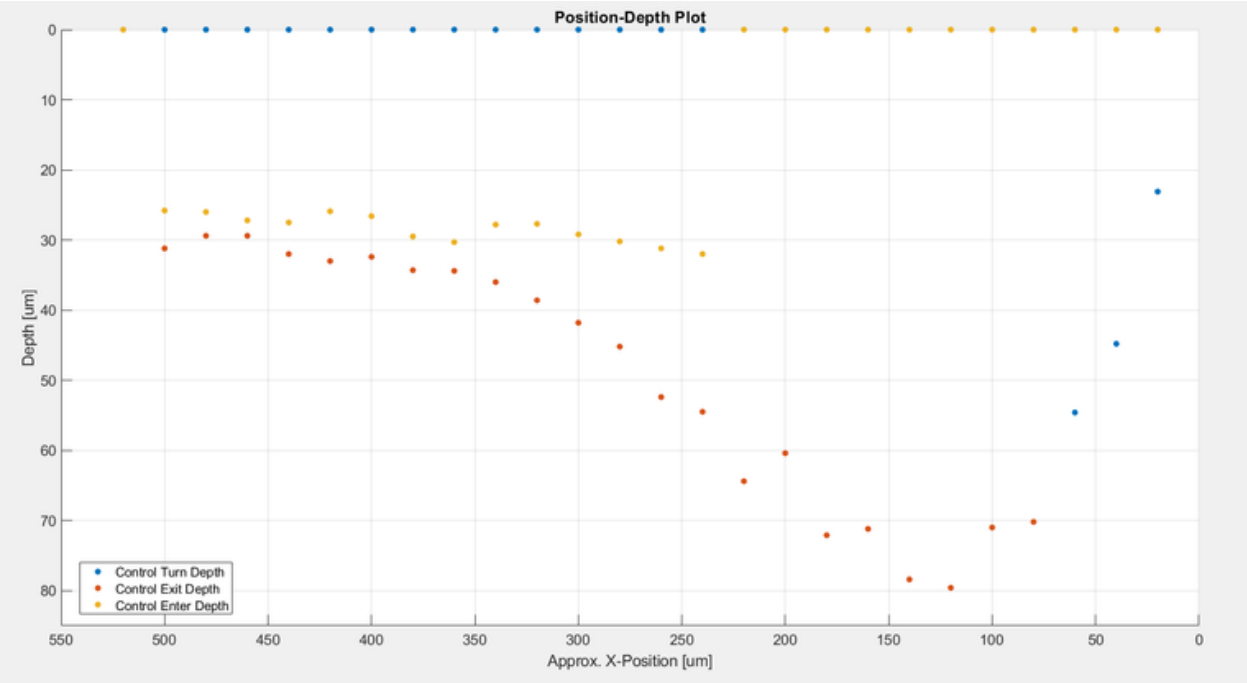

Figure 38: Melt-pool depth per section position for CP VSV1.

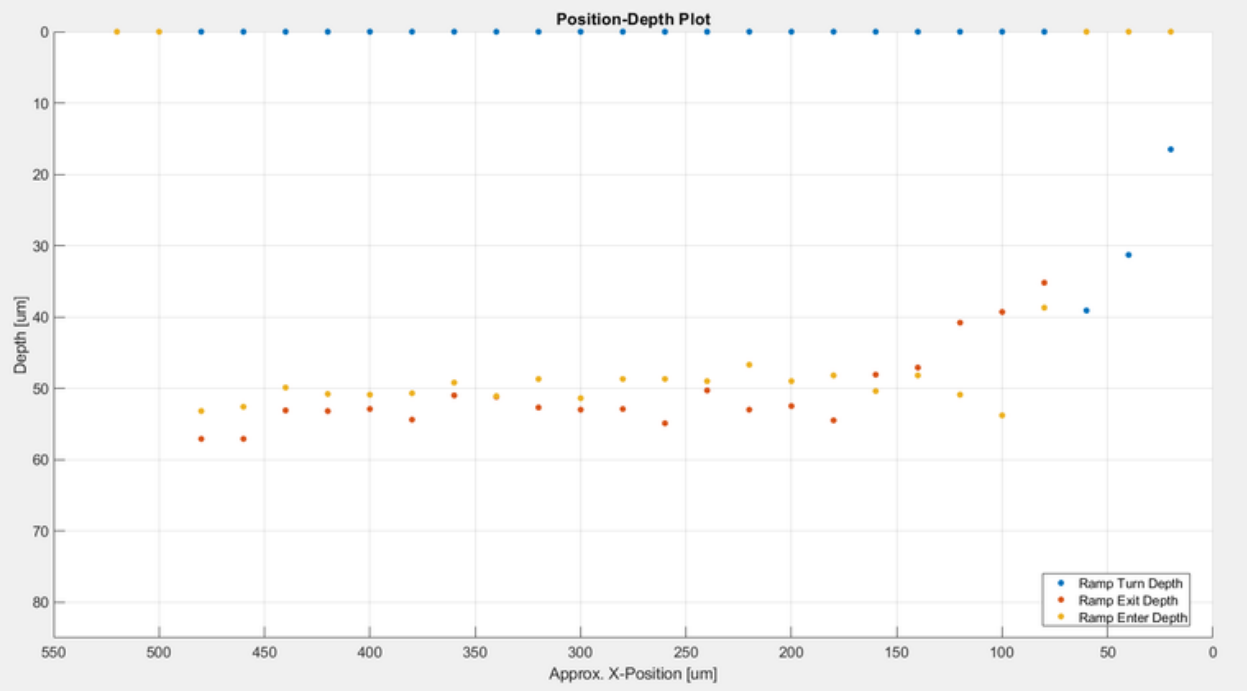

Figure 39: Melt-pool depth per section position for RP VSV1.

Only depth was initially considered as a metric for cross-section analysis. Based on the sectioning geometry, after grinding past the turnaround, 2 distinct melt-pools can be identified and were labeled as enter and exit depth for whether the tracks were pre- or post-turn. The CP scan experienced a large increase in depth ~ 80um, Figure 38, compared to RP's ~ 57um, Figure 39. After the spike, the CP melt retained elevated depths from ~ 120um to ~ 360um. In contrast, the RP melt maintained low variability between the entering and exiting sections.

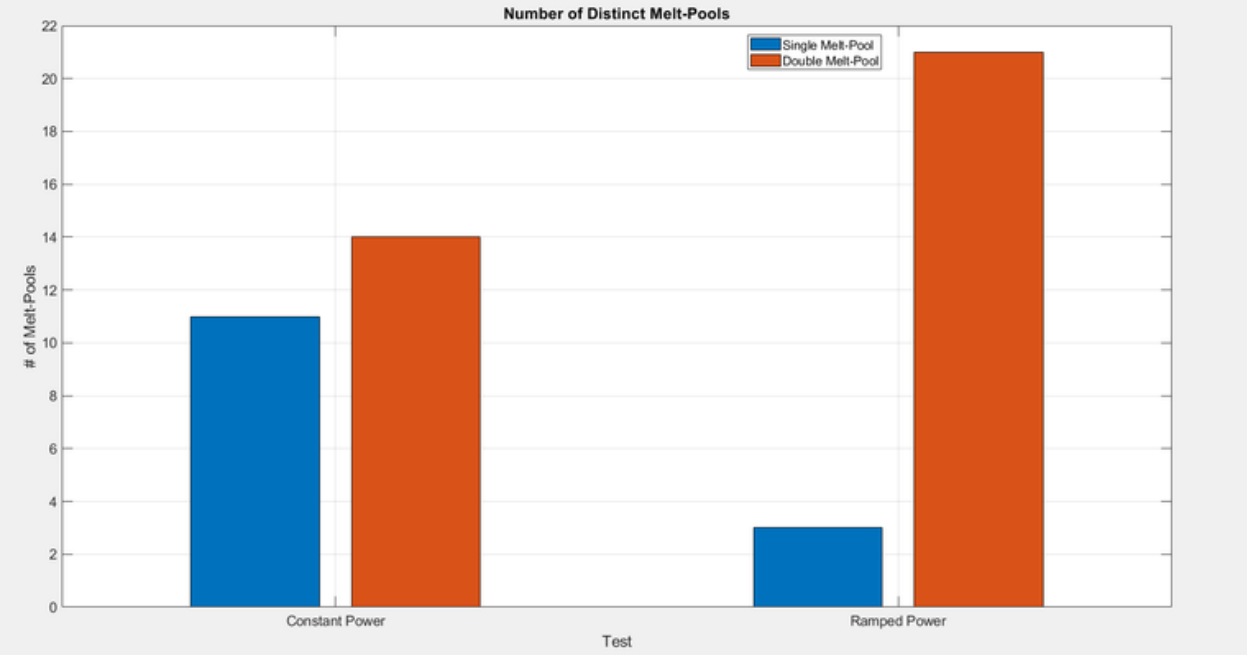

Figure 40: Number of overlapped vs distinct melt-pool pairs VSV1.

The quantity of single vs double melt-pools for the CP scan was 11 to 14 as opposed to the RP scan's 3 to 21, Figure 40. After isolating the first 3 melt-pools, ~ 60um of each track, the CP scan still showed excessively wide melt-pools which would be expected of increased heat accumulation during deceleration. The effect of the overheating was pronounced enough to completely overlap the melt-pool pre-turn over ~ 160um.

Due to potential issues encountered during the metallography process, a secondary reference point was identified. The over melting seen in the CP scans and the initial sections for the ramp show single melt-pools whereas further into the track the RP reveals 2 distinct melt-pools, this was attributed to orientation and section distance. Eventually the CP scan also shows separated melt-pools after exiting the acceleration region at the start of vectors. If the number of visible melt-pools is equal to the number scanned, as mentioned above, there is no reference point to relate surface position to the viewed section. However, if the RP scan shows at least 1 single melt-pool it would indicate that the initial section is not past the turn.

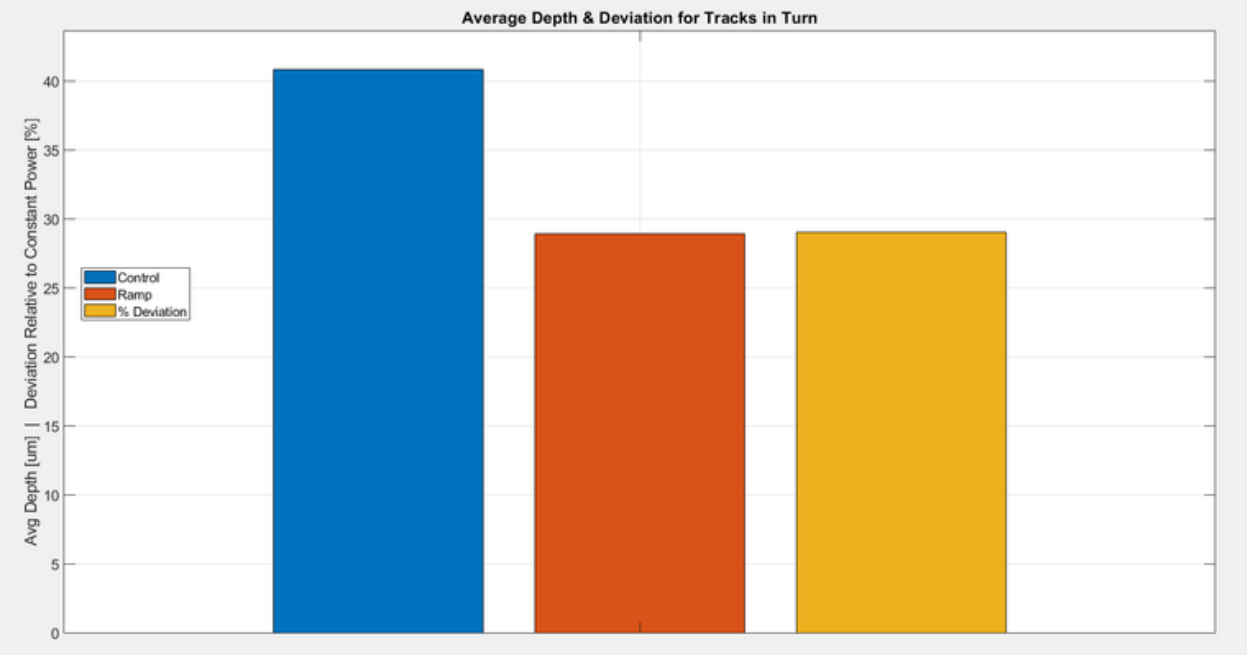

Figure 41: Average depth results for CP & RP melt-pools in the turn VSV1.

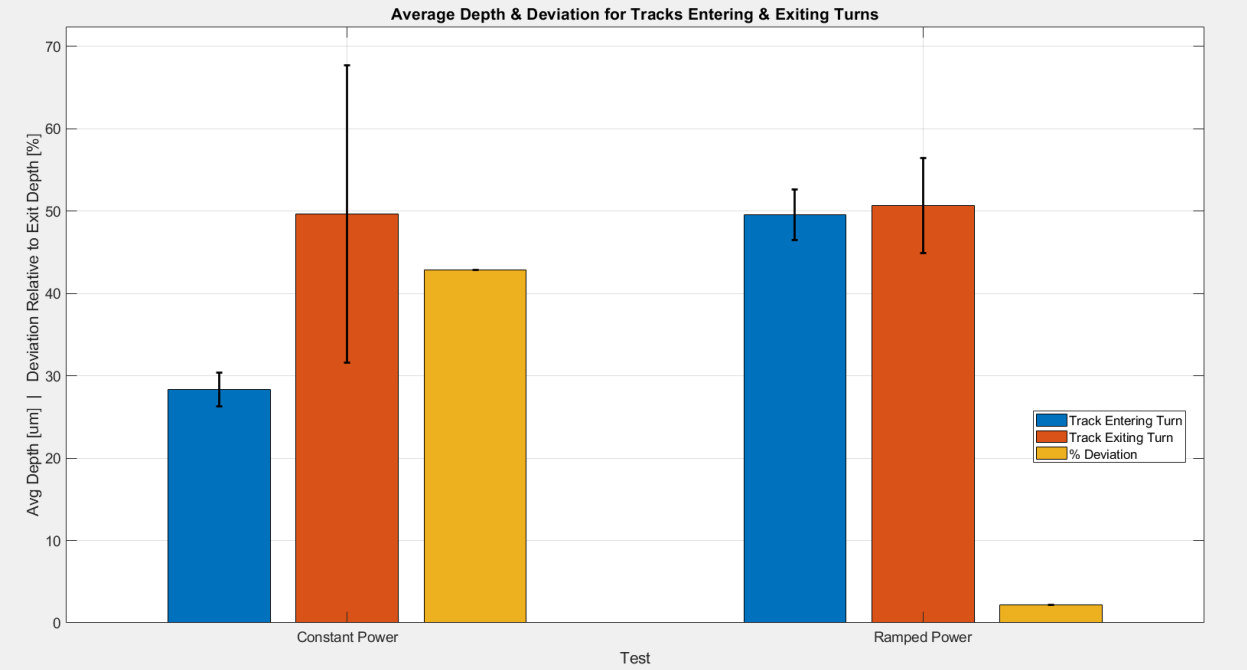

Figure 42: Average depth results for CP/RP melt-pools entering/exiting the turn VSV1.

Considering the small sample size for the melt-pools classified as part of the turn, 3 in either scan, the results should not be considered as impactful as those of the pre/post-turn groups. Additionally, the accuracy of the section with respect to distance can be at most 20um between

melt-pools. It is possible that the first melt-pool for the CP scan could be sectioned up to that distance further into the track than the RP scan, impacting depth results. Even so, the RP scan saw ~ 30% reduced mean depth compared to the CP scan in the turn, Figure 41.

In the case of the pre/post-turn groups for both scans, Figure 42, although discovered post-hoc, the defocused beam in the case of the CP scan would have impacted the depth at all points as seen by the pre-turn average of ~ 30um. However, the high standard deviation, ~ +/- 18um, in the post-turn results, and deviation between both groups suggests that the increasing energy density did impact melt-geometry. Comparatively the RP scan had ~ 40% less deviation between groups and 3 times reduced standard deviation for the post-turn results. Excluding outside factors, maintaining a constant normalized enthalpy relation P/sqrt(V) should result in constant vapor depression depth regardless of what the target depth may be. The metrics discussed above consider the portion of the scanned tracks which undergo changes in velocity, necessitating a power adjustment. The variability in melt-depth moving forward will be used to quantify the effectiveness of the applied power ramps.

4.1.4 Test Issues

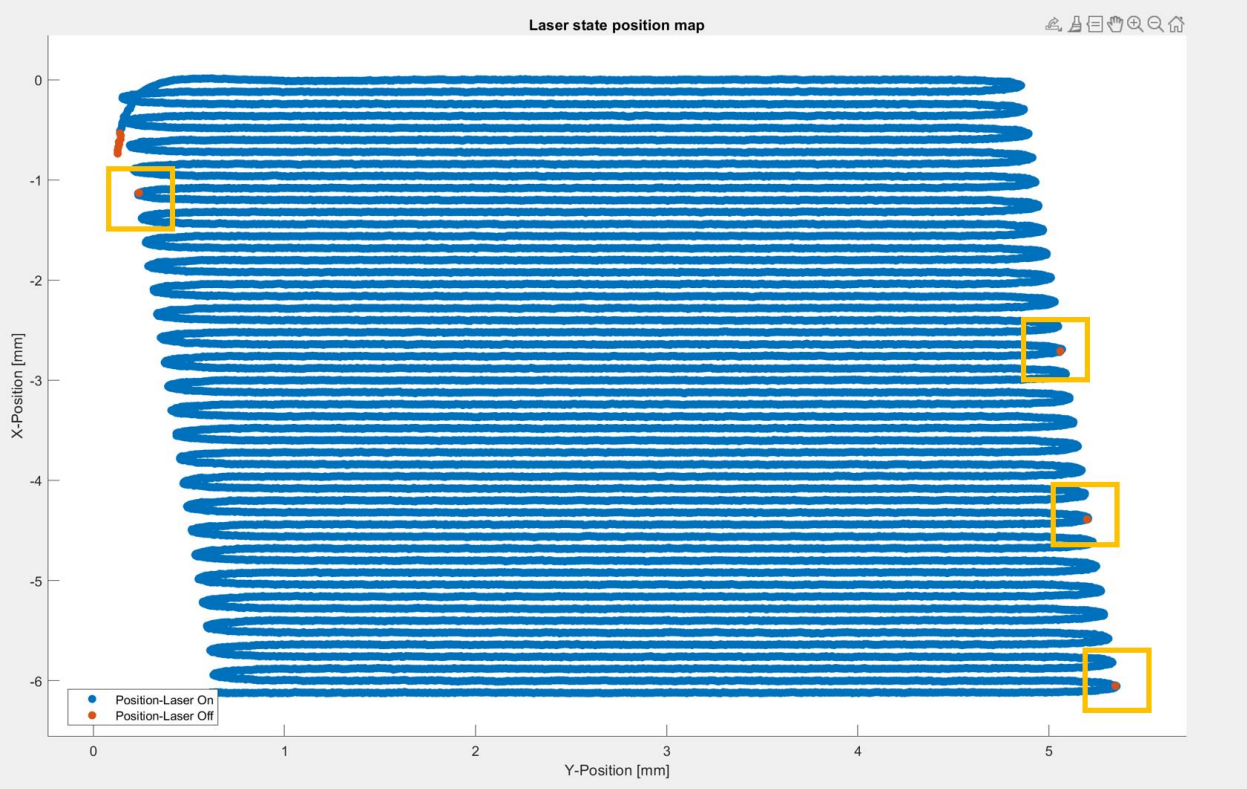

Figure 43: Laser-off points in turns.

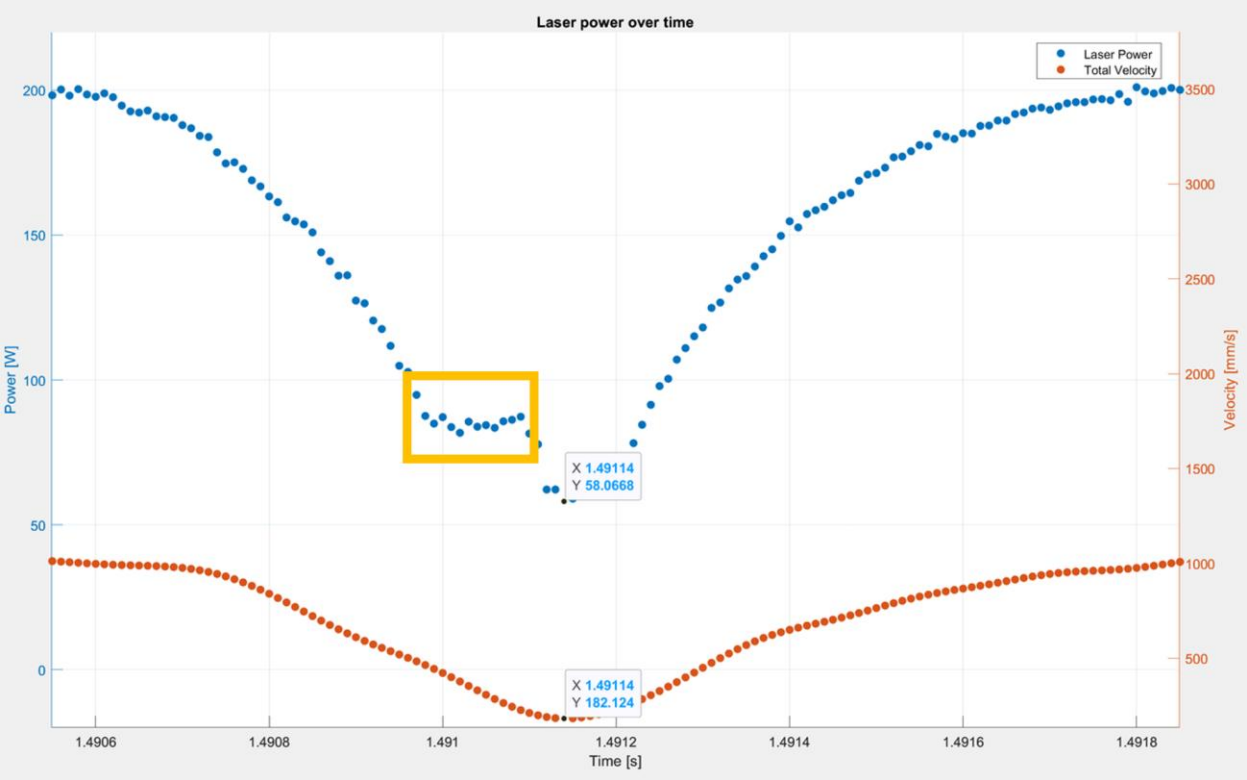

Figure 44: Inaccurate execution of laser power profile in RP scan.

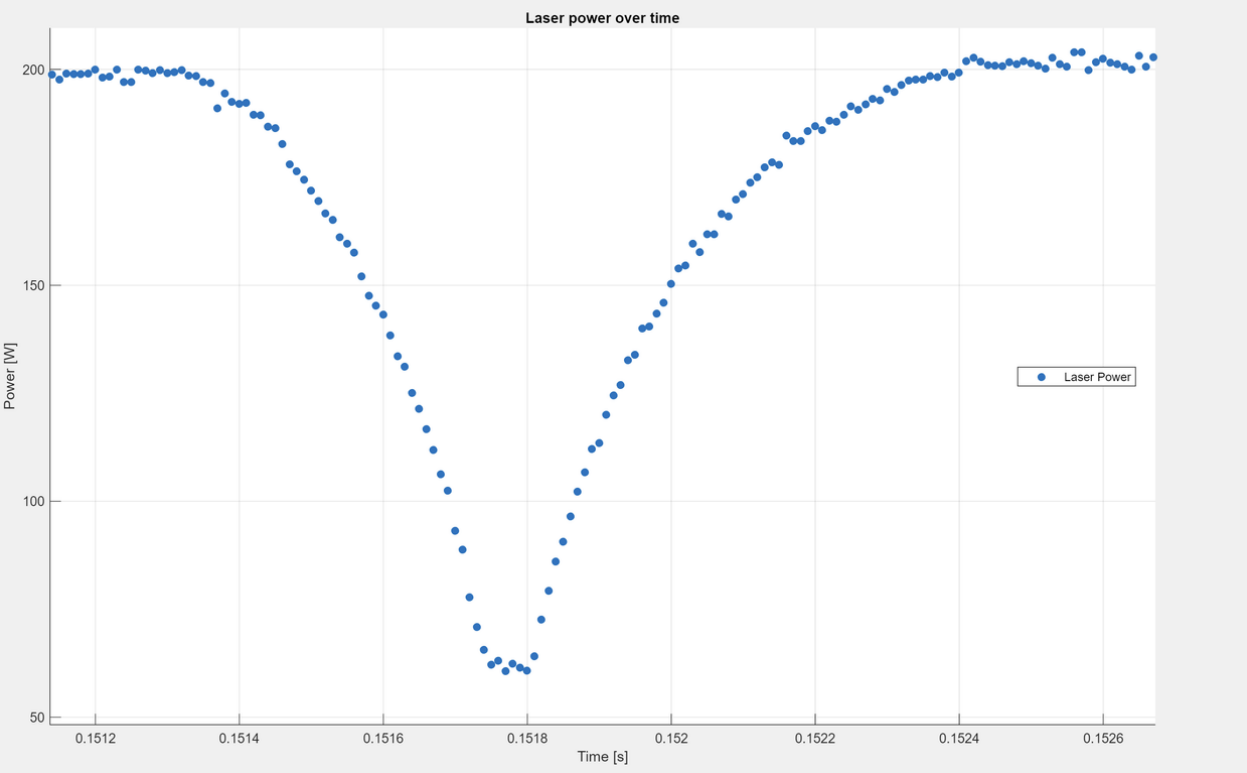

Figure 45: Accurate execution of laser power profile from earlier tests.

However, as noted in the previous sections, multiple discrepancies required additional investigation and the findings necessitated both improvements to the testing process and the build file generation. A delay between data collected from the laser and the scanner was identified and resulted in misaligned data, which was used for analysis, Figures 48-49. To correct for this in future scans, the laser data was shifted forward 80us. Several scattered laser-off points were identified in the turns which may have impacted the resulting melts, Figure 43. Power data was cross-referenced for the times in question but no drop in power was found. Due to the small timescale of the laser state irregularities any conceivable deviation in processing is unlikely to have been detected on the surface or in the cross-section. The miscalculation of enthalpy values is shown in Figure 46, although the values not in a ramp such as in Figure 47 would not be affected.

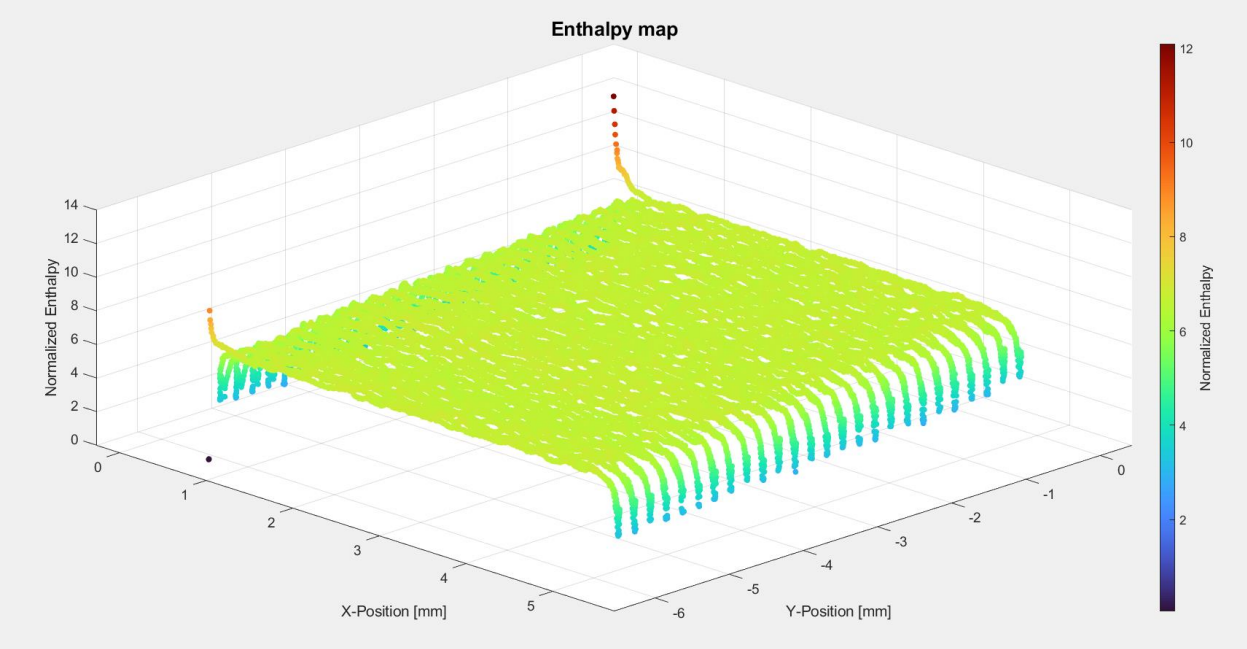

Figure 46: Normalized enthalpy decreasing at ends of vectors for RP scan.

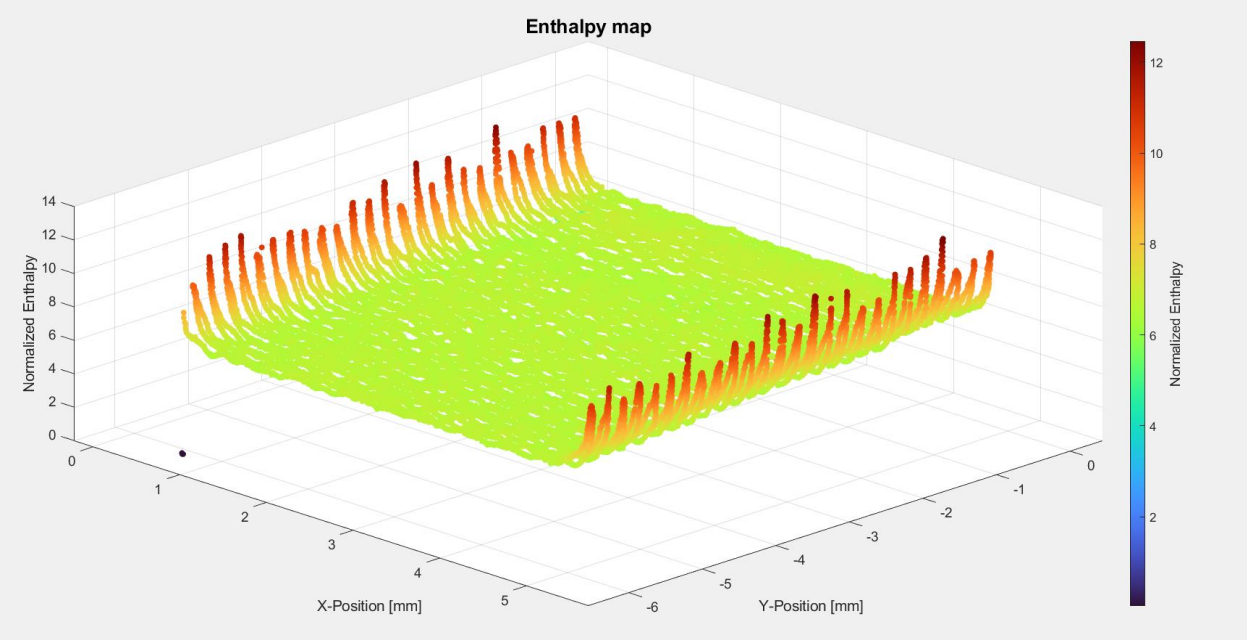

Figure 47: Normalized enthalpy increasing at ends of vectors for CP scan.

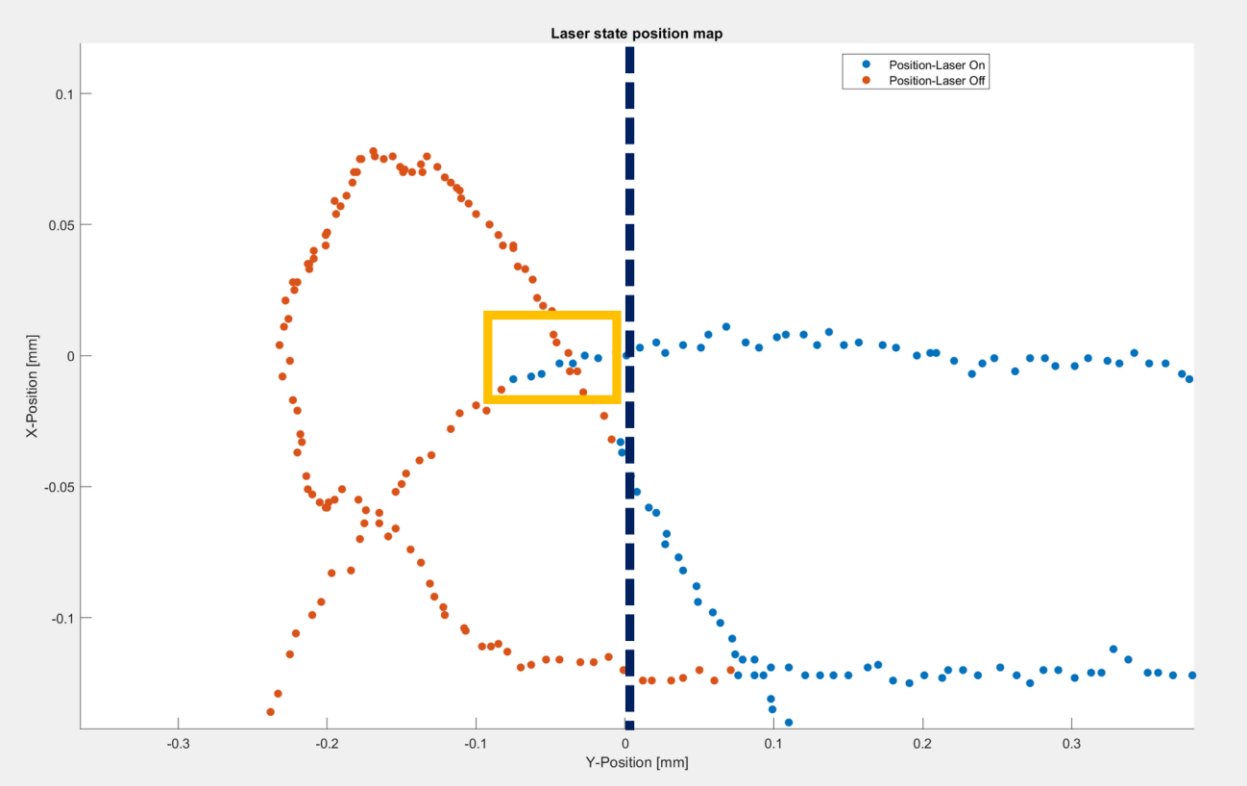

Figure 48: Laser on delay appears to be inaccurate.

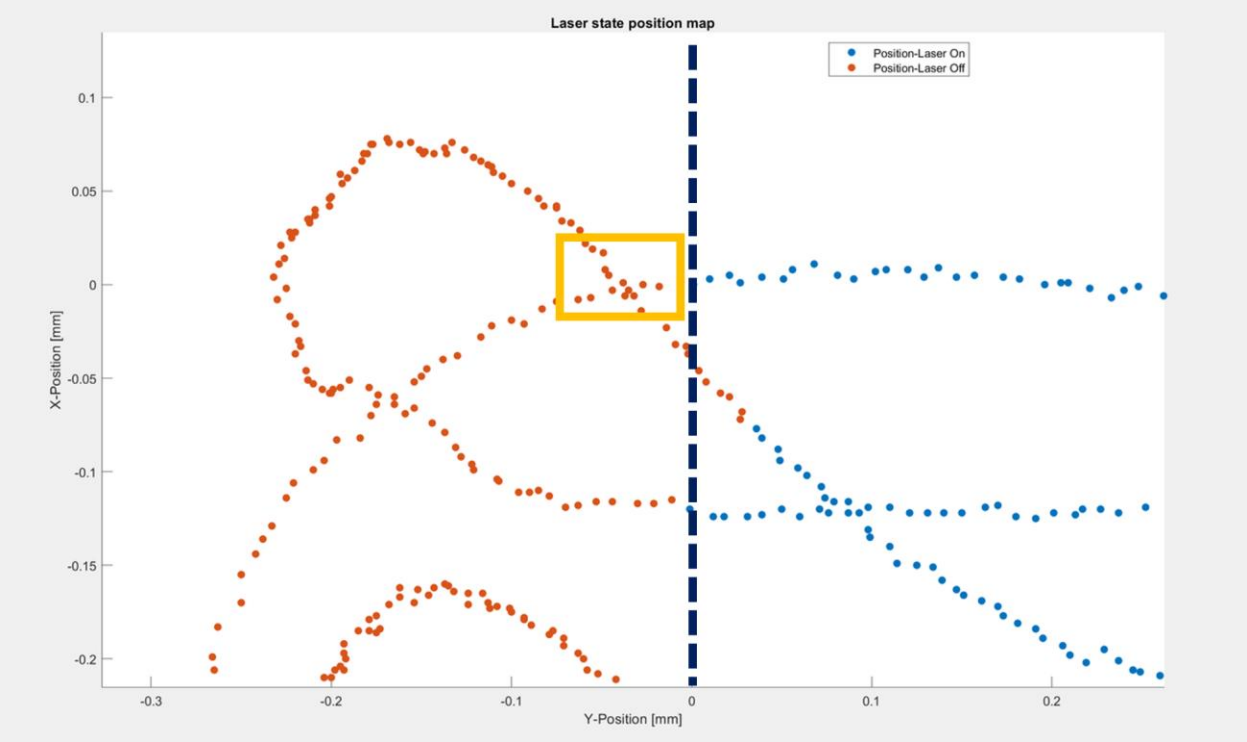

Figure 49: Shifting the laser data set.

A disparity in depth occurred in the first 3 pairs of melt-pools not considered part of the turn for the RP scan. This was attributed to an irregular response in the first half of the power ramp which appeared to have applied higher than commanded powers at instances of deceleration, Figures 44-45. Initially the lower power cutoff limit was suspected to have caused the flatline, but it was verified to be 50W, below the commanded setpoint of 60W. The exact cause was not identified but the response did not appear again in additional testing. The increased track width in the constant scan was attributed to operator/machine error and the results of the second validation scan supported this. Additionally, the sectioning plan was modified so that 2 sections were to be taken for validation scan 3 and successive scans. The steady-state region would be sectioned as well as the ends of vectors which can provide additional benefits.

Due to assumptions made regarding material properties at or above melting temperatures, existing sources were referenced for optimal enthalpy targets for testing. After establishing improved testing procedures and ramp characteristics, the process windows used would be used

to calculate enthalpy values and cross-referenced with earlier sources. From this we can perform testing at various nominal powers/speeds and compare data, evaluating the linear relationship between vapor depression depth and an enthalpy value based on mostly assumed constants. Furthermore, investigation of the required accuracy for various profile parameters such as minimum power steps between micro vectors and effect of temporal resolution for power steps to achieve improvements in melt-pool depth variability is of great interest. Identifying a point of diminishing returns in terms of required control may also help guide optimization of scan strategies.

4.2 Validation Scan V2

A second validation scan set was run to help verify several unresolved irregularities. The same files used in the first set were retested under the same process conditions, with additional attention paid to the discrepancy in track width and RP irregularities. Surface results were compared, but issues with the metallography process for the steady state regions resulted in limited comparisons for the sections at ends of vectors.

4.2.1 Monitoring Results

In the steady state regions, the power and velocity for the first 5 turnarounds were analyzed, Figures 50 and 53. The average power for the RP scan was ~ 201.8W and for the CP scan was ~ 201.5W. The average velocity for the RP scan was ~ 995.4mm/s and for the CP scan was ~ 993.5mm/s. The maximum error from the nominal parameters for the RP scan was ~ 0.9% for power and -0.5% for velocity. The maximum error from the nominal parameters for the CP scan was ~ 0.7% for power and -0.6% for velocity. Although characteristics of the ramped power profile cannot be identified here, in the future running a physical index scan where all points outside of the ramp are set to zero could be used to provide a clearer distinction. As seen in the first scan set, the decrease in power from nominal to the lower limit of ~ 60W was shifted to

align it with the midpoint of the turn where acceleration peaks. Additionally, the same build files were used so no correction to the profile for using total velocity was made, as such, the drop in enthalpy is seen again for the RP scan, Figures 51-52. Similarly, the increase in enthalpy shown for the CP scan, Figures 54-55, is much more evident in the section images after having addressed the defocus error.

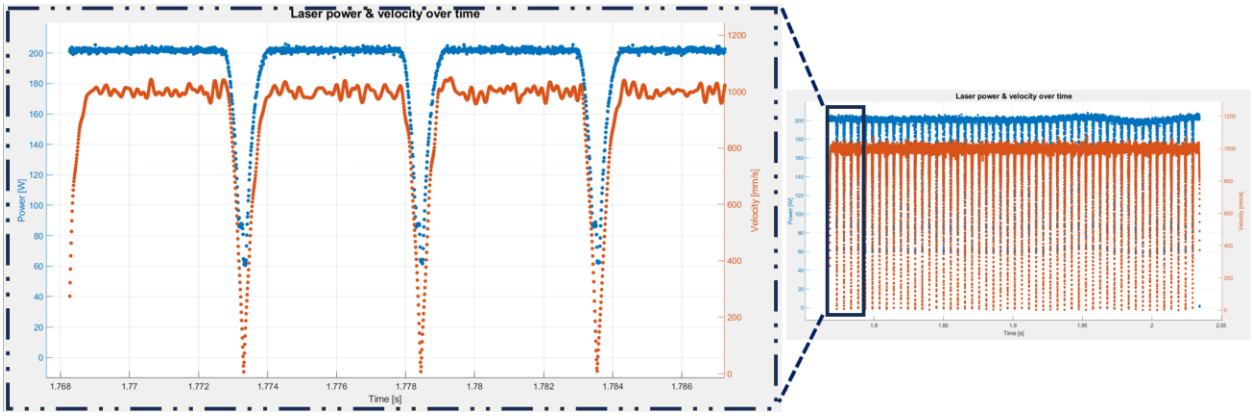

Figure 50: Power and velocity data for RP VSV2.

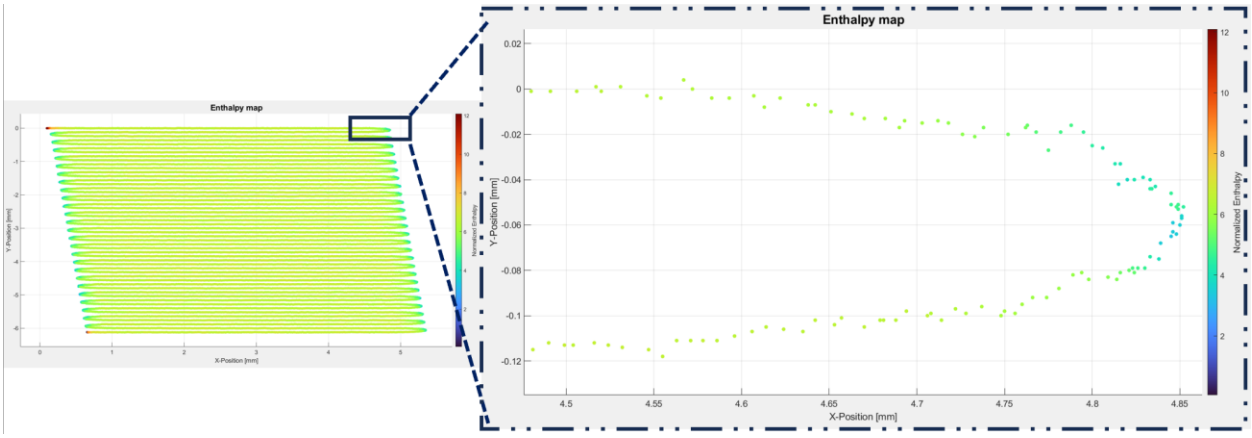

Figure 51: Calculated enthalpy for RP VSV2.

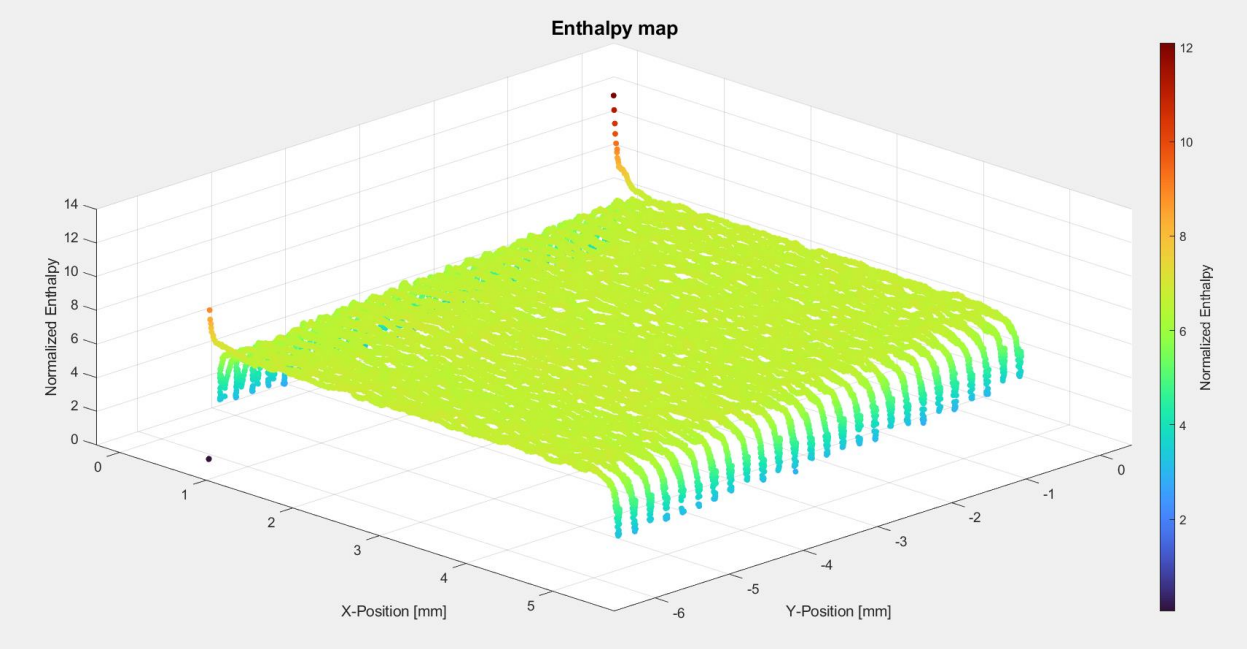

Figure 52: Enthalpy drop at ends of vectors with RP.

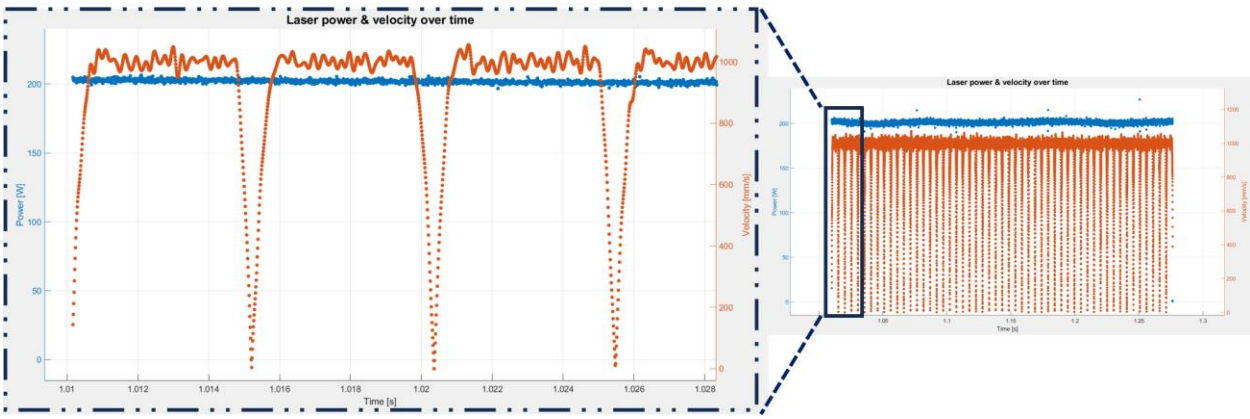

Figure 53: Power and velocity data for CP VSV2.

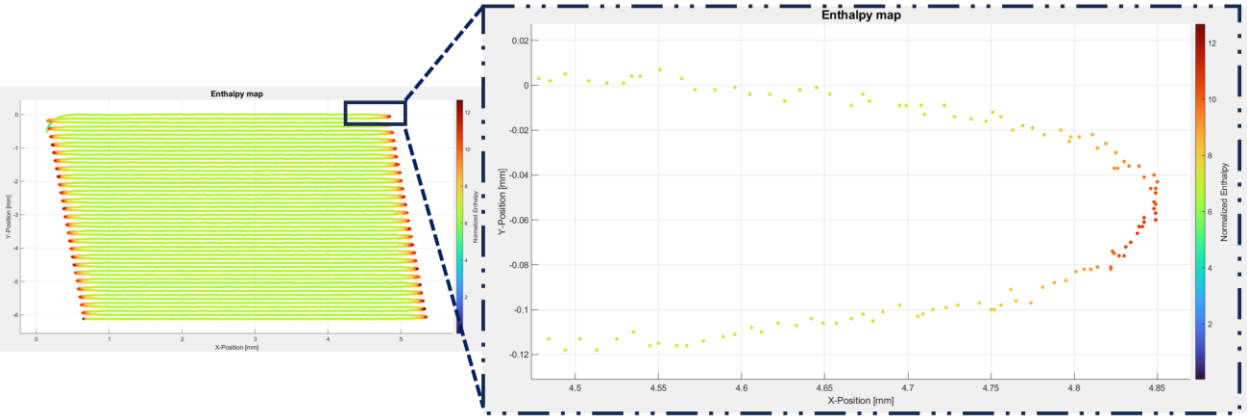

Figure 54: Calculated enthalpy for CP VSV2.

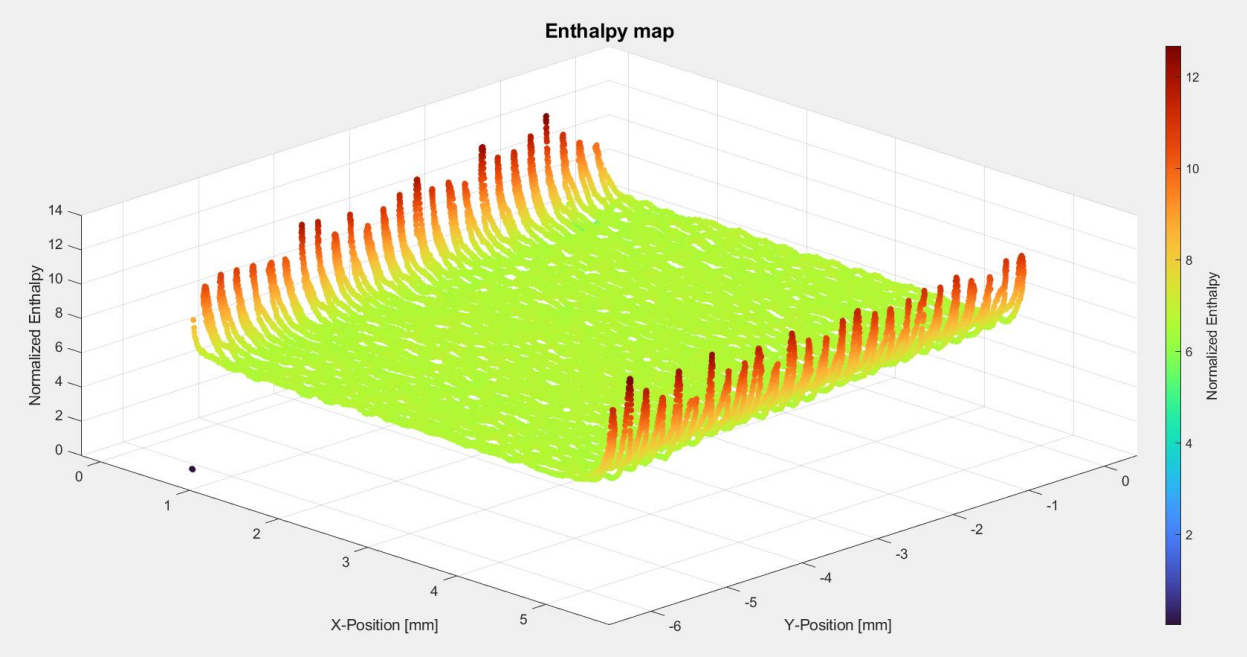

Figure 55: Enthalpy spike at ends of vectors with CP.

4.2.2 Surface Results

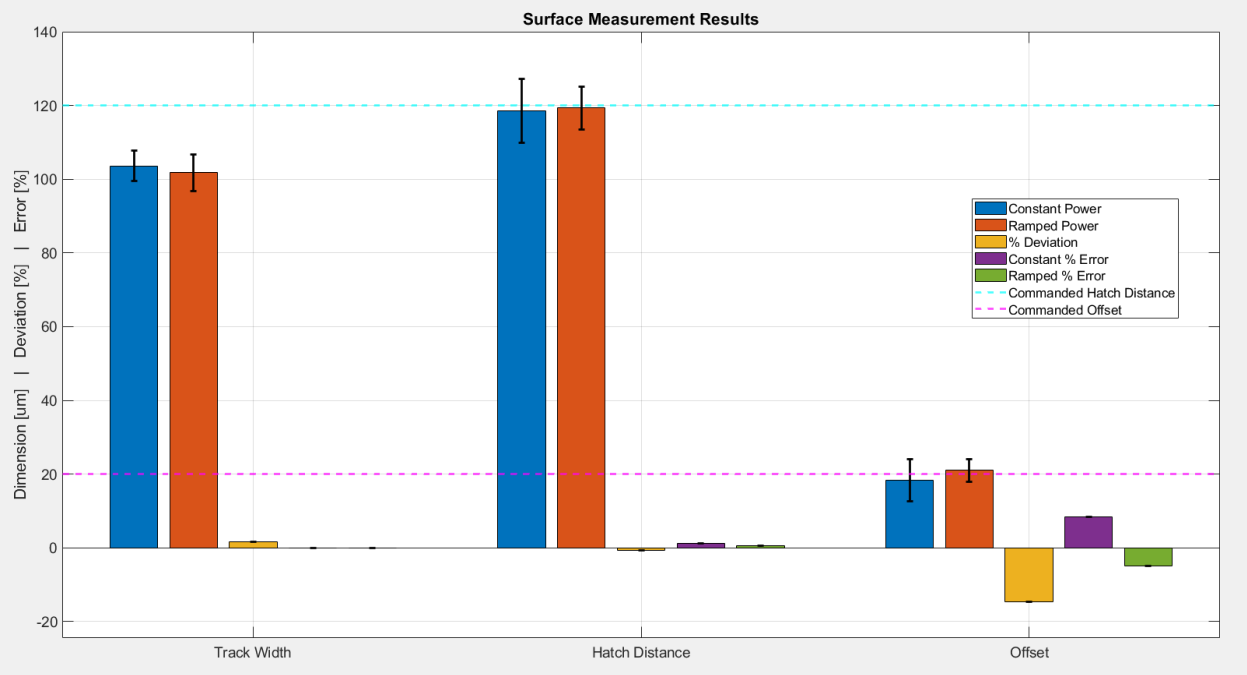

Figure 56: Surface measurement results for VSV2 scans.

Hatch distance for the second set had comparably low error as with the first set, Figure 56. Offset deviation increased from ~ 3.3% in the first scan set, to ~ 14.5% in the second. Any significant deviation from the commanded offset affects the approximate distance of sections between each vector pair. For the purposes of this study, all approximate distances shown in the context of section results are assumed to be within 20um of the true distance simply based on the commanded offset. Track width deviation decreased from ~ 16.4% in the first scan set to ~ 1.7% in the second. This pointed to a resolution of the track width error but was confirmed in the section results.

4.2.3 Section Results

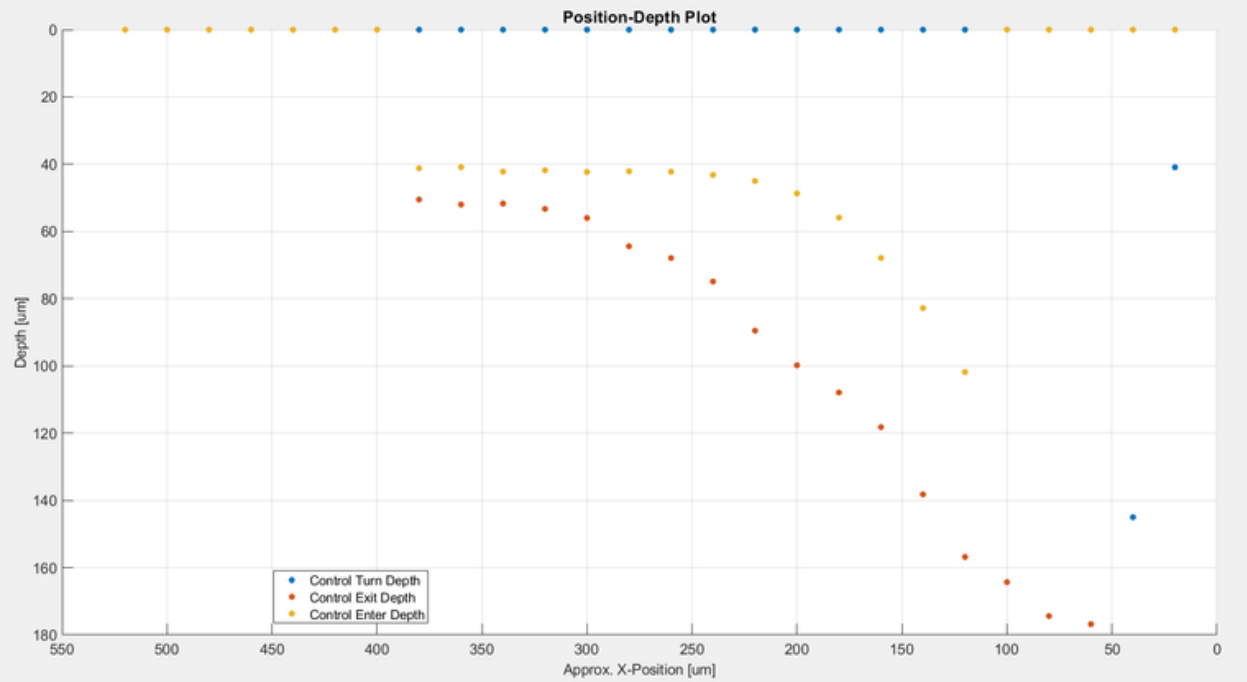

Figure 57: Melt-pool depth per section position for CP VSV2.

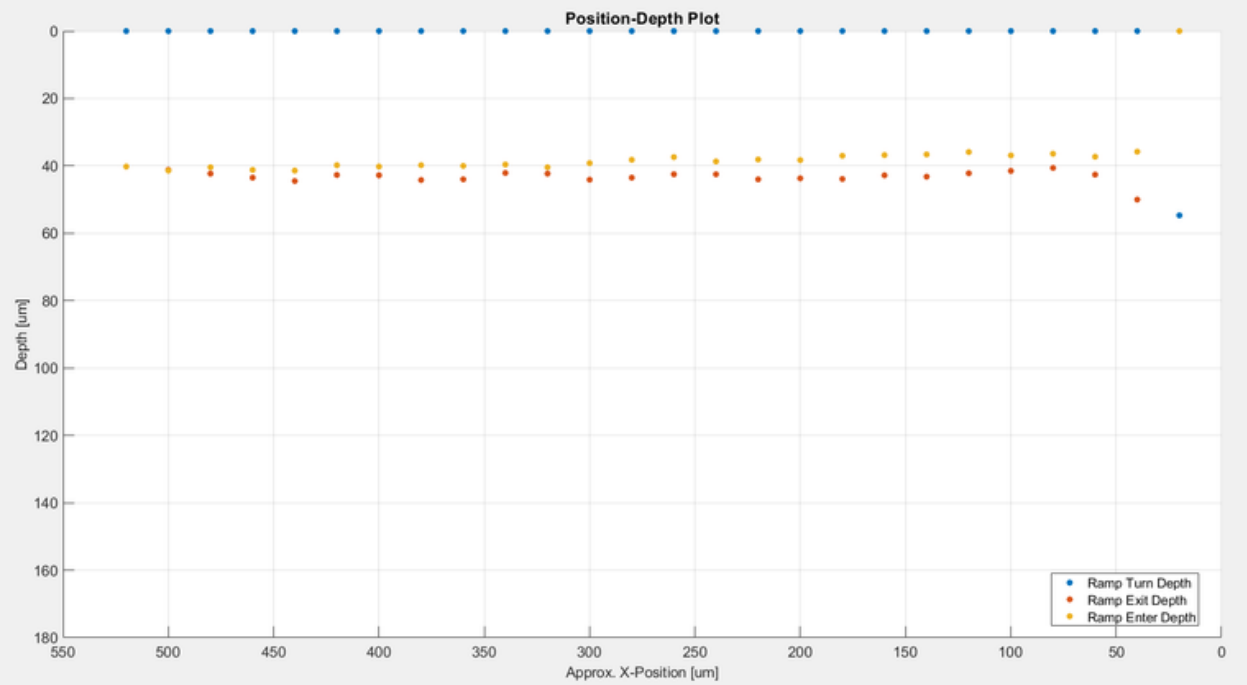

Figure 58: Melt-pool depth per section position for RP VSV2.

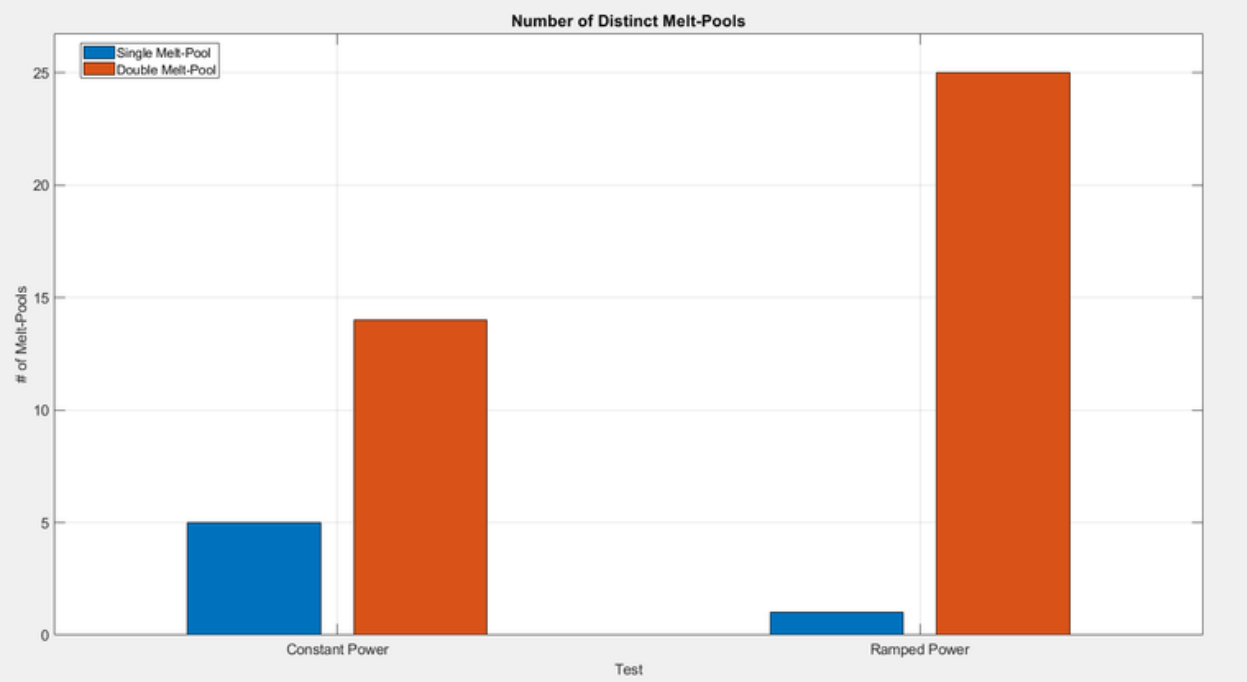

Figure 59: Number of overlapped vs distinct melt-pool pairs VSV2.

The CP scan in the second set, Figure 57, saw a significant increase in depth relative to the RP scan, Figure 58, and even compared to the CP scan of the first set. A maximum spike of ~ 176.8um was identified for the first melt-pool post-turn for CP compared to ~ 50um for the RP. Due to complications during sectioning, only 19 pairs of melt-pools were captured for the CP scan whereas all 26 were captured for the RP scan. Additionally, the steady state section for CP was over grinded pushing it to the other end of the vectors. However, considering the pre-turn depths for CP, and the trend in RP, it's likely that CP would have eventually normalized to ~ 40um in depth as it approached steady state conditions. From what we can see however, the melt-pools in the CP scan retained elevated depths from ~ 60um to ~ 300um post-turn and from ~ 200um to ~ 120um pre-turn with additional melt-pools being identifiable compared to the first scan set. This was due to the pre-turn melt-pools over-melting enough to still be visible after the post-turn group overlapped their partner. By contrast the variability in the RP scan was greatly reduced. The quantity of single vs double melt-pools for the CP scan, Figure 59, was 5 to 14 as opposed to the RP scan's 1 to 25. Similar to the first scan set the CP scan showed signs of over-

melting from the increase in width to the point of overlapping the partner track for a distance of ~ 160um.

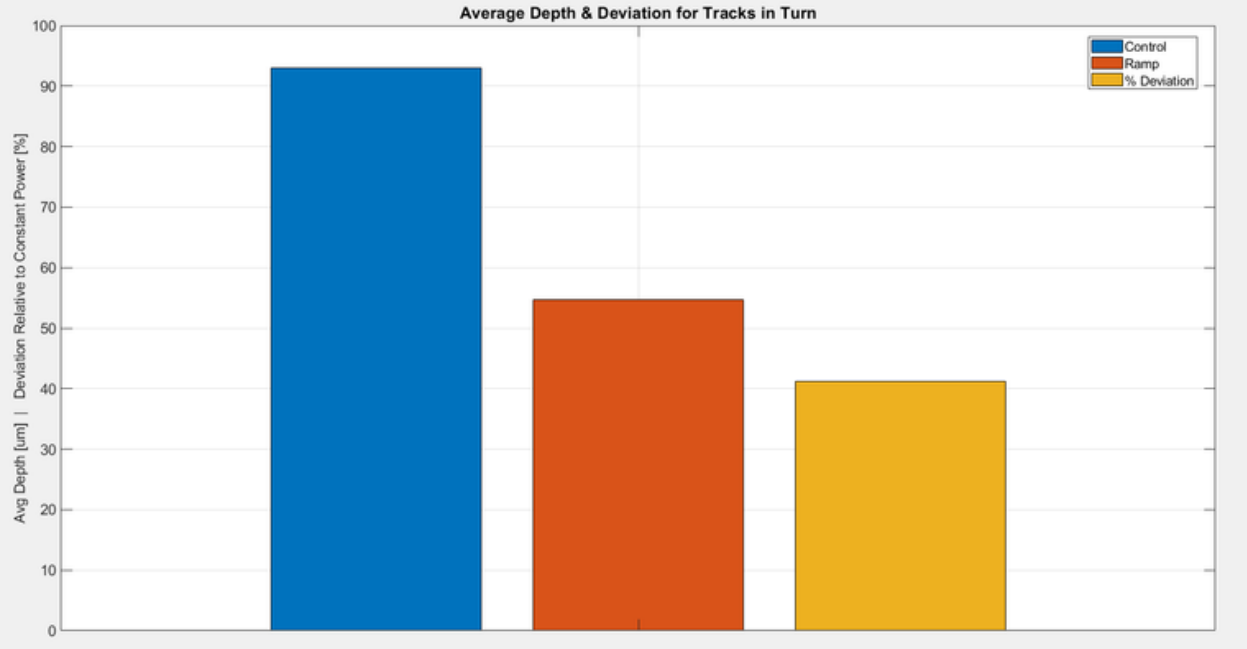

Figure 60: Average depth results for CP & RP melt-pools in the turn VSV2.

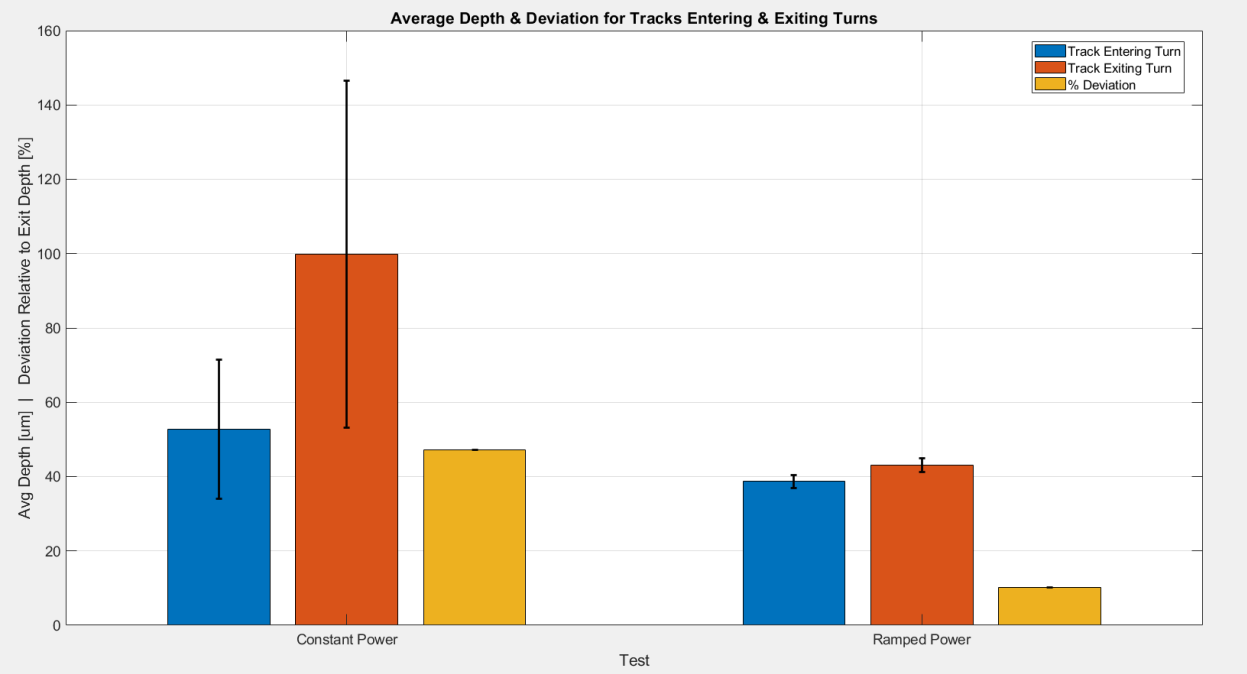

Figure 61: Average depth results for CP/RP melt-pools entering/exiting the turn for VSV2.

The findings for mean depth pre-turn, in the turn, and post-turn were in line with the first scan set's results, Figures 60-61. In the turn the ramping power resulted in a reduction in depth of over 40%. Pre- and post-turn mean depth for RP was ~ 38.7um and ~ 43.1um respectively. The mean depths for both groups for CP was ~ 52.7um and ~ 99.8um. Comparing the deviation between pre- and post-turn melt-pool groups, RP saw an ~ 37% reduction, ~ 10.2% for RP versus ~47.2% for CP. The large spikes in depth in the CP scan also severely impacted variability within each group, resulting in a standard deviation of ~ +/- 18.7um and ~ +/- 46.7um for pre- and post-turn melt-pool groups respectively. By contrast the RP scan possessed relatively low standard deviation, ~ +/- 1.8um for pre- and post-turn melt-pool groups respectively.

CHAPTER 5: DISCUSSION

5.1 Discussion

The major products of this work are primarily in the developed workflow for:

1. Defining scan paths that provide applications for inter-vector power ramping.
2. Processing scan data to characterize transient acceleration events at ends of vectors.
3. Integrating kinematics as the basis for defining an adjusted power profile.
4. Developing a program to provide the printer with simple geometry and power data.
5. Verifying scan strategy execution with in-process monitoring data and plate scan surface/cross-section morphology.

There is potential for increasing control on LPBF systems in scenarios where typical functionality may provide suboptimal processing. The focus of this study was demonstrating a constant build mode with the intent of maintaining melt-pool morphology during transient events. This was compared to a constant power scan which served as a baseline for insufficient parameter control. The potential applications may be limited for this specific mode given several limitations. By attempting to apply ramping on a commercial system, the avenues of approach were constrained by system hardware and software. Although adjustment of scanner delays may provide additional applications for these ramps, adjustment of skywriting parameters was used for the type 1 geometry as described in chapter 4. While the main benefits may be reduced build time and melt-depth with reduced variation at ends of vectors, the change in vector length and melting during the turn may not be desirable depending on scan geometry.

Should this geometry be applicable to a particular design, any system which provides a means of the following can benefit from this workflow. Monitoring of laser power and scanner position is required at a minimum, in addition to options for adjusting power between vectors or groups of vectors. The system would also require a means of adjusting skywriting parameters for type 1 geometry specifically, or scanner delays as a means of related but distinct ramping options. Calculating velocity from position data and filtering enabled a means of characterizing

the scanner's motion at high resolution timescales. Relating power to velocity as based in normalized enthalpy for controlling laser power provided a measurable difference between ramps and the baseline. The program enables the definition of any theoretical power profile constrained only by the hardware specifications and the required complexity of the geometry. Monitoring the laser and scanner data provided sufficient means for identifying irregularities, validating execution of the commanded strategy, and the plate analysis enabled a means of quantifying ramp effectiveness via multiple metrics.

CHAPTER 6: CONCLUSION & FUTURE WORK

6.1 Conclusion

Single layer plate scans of multiple pairs of equivalent vectors were performed on a commercial AM system. The geometry defined as type 1 reduces the extension time parameter for skywriting to 0us which results in the skywriting motion being shifted into the mark section, enabling a constant build mode. Monitoring data from the scanner and laser was used to produce a nonlinear approximation of the scanners x-component velocity profile. This was integrated into a power calculation which utilized a relation of power to speed derived from the normalized enthalpy analytical model. Lastly a program was developed to use the power calculation to generate geometry and power ramps in the system's accepted build file format.

Results from both scan sets showed similar findings where the maximum depth identified in the group of melt-pools exiting the turn was higher for CP scans in each validation scan set. The standard deviation of the same group was higher for the CP scans and the deviation between the group of melt-pools entering the turn vs exiting the turn was also higher for the CP scans. Note that this does not represent a statistically significant difference in means. Multiple issues identified during and after testing impacted results of the first scan sets processing and the second scan sets section results, and each scan set had a sample size of 1. However, considering these findings only as early observations, the need for improvements for the discovered issues and for additional testing can at least be established.

In summary, the findings from exploratory experiments were in line with the hypothesis. Implementation of a constant build type scan strategy on a commercial LPBF printer was explored. The application of ramps at ends of vectors appears to have reduced variability in melt-geometry compared to CP scans. This may provide additional opportunities for tailoring melt-pool geometries to compensate for surrounding factors or achieve specific properties. A methodology for processing basic monitoring data and leveraging it to define a power profile for commanding ramps was developed. Issues associated with initial tests such as misaligned data

collection, inaccurate approximations, operator errors, geometry limitations, etc. were investigated and improvements made for future tests. Applications for the proposed scan strategy may be more limited given the incapability to mark exact geometry and the lack of delays between marking. But the defined methodology has been applied to another scan strategy under development which if improved will supply more utility and may provide an alternative to standard skywriting. Further applications include targeting defects prone to formation where adjusted energy density cannot be assigned using commercial AM software.

6.2 Future Work

There are numerous unexplored areas of research closely related to the process outlined in this body of work:

6.2.1 Type 2 Geometry Development

Another application for the CLI+ functionality is as an alternative to the skywriting modes common to LPBF systems. Skywriting has 2 intrinsic flaws related to material response and the instantaneous change in energy input. At starts of vectors there is insufficient time for the melt to develop, after the laser has switched to the on state, resulting in lower melt-depth at starts of vectors. At ends of vectors, after the laser has switched to the off state, the solidification rate of materials exceeds the time required for molten material surrounding the melt-pool to flow back into and fill the cavity. The latter results in a “frozen depression” which from surface images is noticeable due to out of focus regions at ends of vectors for 2D stitched images. Section images show reduced melt-pool height, below not just entering melt-pools but also the reference line of the plate. While these microfeatures represent a small portion of process limitations, the potential impact of anomalous events during processing of large build volumes may measurably impact mechanical properties. Mitigating these features have been accomplished in existing studies but implementing this improvement on a commercial system

using the developed process may aid in incentivizing continued expansion of control for operators and researchers.

As discussed in type 1 geometry development, the skywriting extension time was reduced to enable constant scanning while velocity changed to test ramp effectiveness. However, this cannot be compared to standard skywriting since for the latter velocity is commanded constant over the entire scan and the geometry is accurate to commanded. When disabling skywriting, scanner delays can be modified to perform equivalently, in the mark section. It was found that the polyline chains used to discretize geometry were still considered a single mark vector, so setting mark and jump delays to a sufficiently high value enables acceleration at starts/ends of vectors to a complete stop. That constitutes the only change between skywriting, and this proposed type 2 geometry. If a power ramp is applied over the acceleration regions at starts/ends of vectors, energy density relations can be made nearly constant, and time spent scanning at starts/ends of vectors can be extended which should mitigate the targeted features.

Several challenges discovered in an initial scan still need to be overcome before additional testing can be performed to validate these hypotheses. The applied ramps appear to require modification compared to those for type 1. The issue lies with any points that are above the commanded power or below the minimum power limit. In type 1 tests, these were set to the limit and the profile was shifted such that the minimum velocity and power points were aligned. Shifting with type 2 beyond the end/start of vector points is not possible due to the behavior of delays and laser commands. No changes in laser commands can occur during a scanner delay so any shift from the end of vector is instead applied at the start of the next vector which is not ideal. Additionally, laser on/off delays do not appear to affect CLI+ assigned powers which results in misaligned laser and velocity profiles, nonconstant energy density relations. Elimination of points outside of bounds i.e., reducing the length of applied profiles, appears to improve the issue but additional work is required before the ramp characteristics are of comparable quality to type 1 scans.

6.2.2 Validation Scan V3

Multiple improvements have been made to produce more accurate approximations in addition to modifying geometry for the skywriting alternative as described in section 6.2.3. The deceleration and acceleration profiles are now considered separate since it was found that the deceleration magnitude exceeds the acceleration. The identification of bounds in velocity profiles has been automated and tests for gathering data for the approximation had increased numbers of vectors. Compared to the first approximation where a single vector was used and compared to other profiles to determine goodness-of-fit, the profiles over a set of 200 vectors were identified and averaged to help compensate for slight deviations of the mechanical system and residual noise. This came at a cost, as the bound setting still relies on a threshold value which is meant to minimize misidentification of inflections after acceleration has begun. The identification of different bounds between vectors affects not just the upper/lower amplitudes but also the length of time for that vectors profile. Additionally, the length of ramps was found to have significant effects on approximation accuracy, this was set to the minimum of the set to not extrapolate data and produce a single equation. Each step for data processing will be scrutinized to improve accuracy.

Other geometry modifications included increasing the number of vectors to 160, reducing x-offset to 12.5um, and increasing vector length to 15mm which provides additional clearance for metallography processing, and should help prevent overgrinding. The target sectioning distance is based on the approximated ramp length. Adjusting the number of vectors and x-offset is used to ensure the target section matches ramp length. Non skywriting laser delays were optimized, although as noted they do not appear to execute appropriately in combination with CLI+ laser power indices and values. The skywriting behavior executed for the scan path of type 1 geometry was determined to be irregular, the system was effectively taking 2 separate 90 degree turns, rather than attempting a 180-degree turnaround. By separating successive polylines into separate vector groups, the behavior equivalent to standard skywriting was executed.

Scans for this set include initial tests for type 2 geometry using an updated perpendicular sectioning plan to include two separate sections at ends of vectors, and one in the steady state region. A skywriting control scan to compare to type 2 geometry. Marking of a physical index test where all points excluding the ramps have power set to zero to verify ramp characteristics in monitoring data and surface position. CP, constant velocity (CV), and RP tests for type 1 to check effectiveness of sectioning geometry improvements. The CP and velocity scans are used to compare to RP since it is not considered an alternative to skywriting and are meant to effectively show the upper and lower bounds for processing, we seek to avoid by applying RP profiles.

6.2.3 DOE

A design of experiments is being developed in preparation for confirming hypotheses established for type 1 and 2 geometries. For type 1 geometry the hypothesis will be whether a group of ramped scans possesses a statistically significant difference in mean depths when compared to a CP or CV scan. Additionally comparing the variability in the group of exiting melt-pools and the deviation between entering and exiting groups will be considered as well. For type 2 geometry the hypothesis will be whether a group of ramped scans possesses a statistically significant difference in mean depth and height at ends of vectors when compared to a standard skywriting mode. Additional response variables under consideration include: melt-pool width, height, area, and shape.

Options for a simple paired T-test with a ramp vs no ramp condition or a more complex fractional factorial design are being assessed. Although defining the applied ramp as a binary factor would simplify the design, a broader test suite may help guide and accelerate improvements made to the ramps. Testing of different levels of ramps, geometries, energy density relations, processing for monitoring data, and materials are being considered. X-component velocity was utilized for all testing but use of total velocity for approximations and

bound setting, degree of approximation, and additional parameters all influence produced ramps. In addition, there are many potential sources of noise which must be accounted for.

6.2.4 Power Mapping

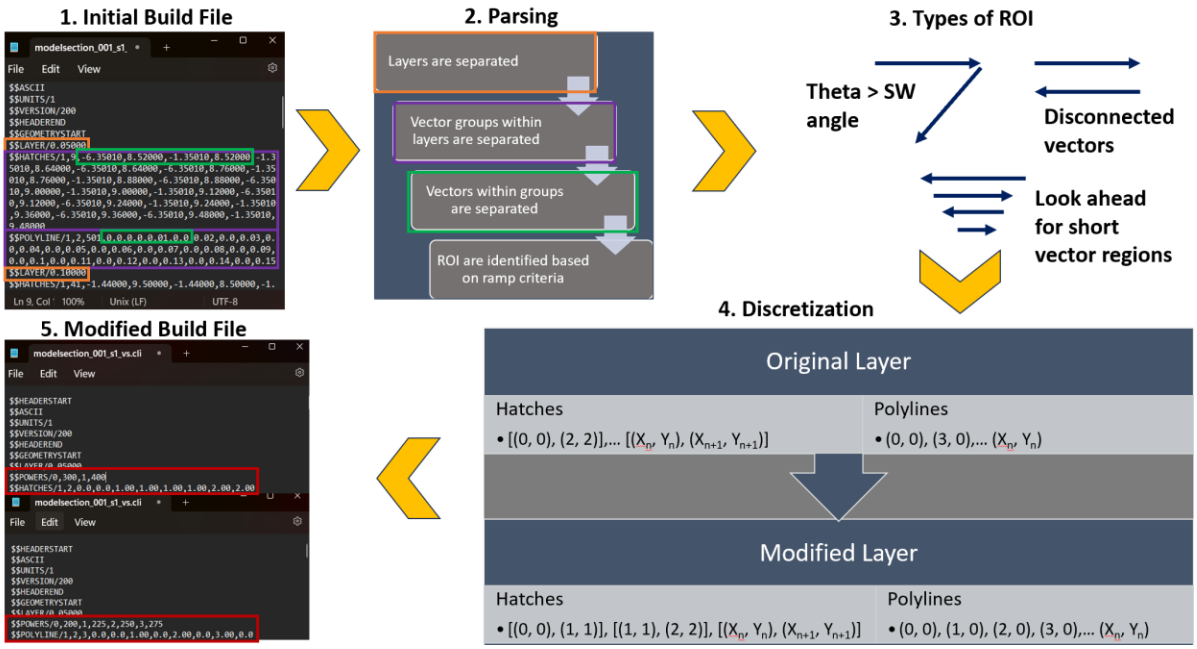

Figure 62: Power mapping concept process map.

Currently, the steady state geometry for scans is calculated in addition to the power ramps. If trying to scale this process to multi-layer tests or more complex geometry, it may be beneficial to merely modify files produced through commercial AM software. Figure 62 details a simplified concept for producing modified build files. Geometry is sorted by layer, vector group, and then ROI can be identified based on the coordinates between 2 or more vectors. If vectors meet a condition such as having an angle exceeding the SW angle, where SW would normally be applied, the geometry can be discretized, a ramp inserted, and then data rewritten.

6.2.5 Speed Control

CLI+ includes additional functionality that has not been explored including options for control speed. This could greatly improve the current work since the power calculation relies solely on a relation to velocity and is limited by inaccuracies in approximating velocity profiles set by default acceleration. Controlling velocity may help simplify ramp profiles and consequently material processing in ROI. But a limitation is that speed can only be set for a vector block rather than by vector indices like power. This would mean separating individual micro vectors into vector blocks with power and speed assignments which would impact file size and program execution time. While data storage and program execution have not been factors of consideration up to this point due to simple single layer geometry, scaling this process will require that.

6.2.6 Integrating Additional Parameters

The summary of work has been accomplished using only laser and scanner data without in-situ process monitoring data. The Aconity MIDI+ has 2 integrated pyrometers for measuring surface emissions and there are options for utilizing off-axis thermal imaging as well. Using options such as these for measuring and integrating additional parameters for regulating process controls e.g., temperature, may further homogenize melt-geometry during transient events. Additional factors which have not been considered include time between successive scans. Up to this point the use of constant energy density relations was the target, but these relations may lack additional considerations. In the case of immediate turnarounds or regions with successively shorter vectors, the time between scans at the ends is lower compared to midpoint of vectors, resulting in less time for preheated regions to cool. In those cases, the energy density used at ends of vectors may need to decrease for successive scans relative to the first vector and steady state regions. Interestingly, the issue discussed in section 4.1.4 regarding the incorrect use of component velocity, resulting in lower power values being assigned relative to a total velocity

calculation displayed a drop in normalized enthalpy. Although accidental, that may provide a more accurate representation of power profiles which consider additional factors as described.

REFERENCES

1. J. P. Oliveira, A. D. LaLonde, & J. Ma, “Processing parameters in laser powder bed fusion metal additive manufacturing,” *Materials & Design*, vol. 193, 2020, doi: https://doi.org/10.1016/j.matdes.2020.108762.
2. M. Vlasea, B. Lane, F. Lopez, S. Mekhontsev, and M. Donmez, "Development of powder bed fusion additive manufacturing test bed for enhanced real time process control," Solid Freeform Fabrication Symposium, Austin, TX, pp. 527-539, 2015.
3. H. Yeung, K. Hutchinson and D. Lin, "Design And Implementation Of Laser Powder Bed Fusion Additive Manufacturing Testbed Control Software," 32nd Annual International Solid Freeform Fabrication Symposium – An Additive Manufacturing Conference, Austin, TX, US, pp. 1455-1473, 2021.
4. A. Martucci, F. Marinucci, A. Sivo, A. Aversa, D. Manfredi, F. Bondioli, P. Fino and M. Lombardi, "An automatic on top analysis of single scan tracks to evaluate the laser powder bed fusion building parameters," *Materials (Basel),* vol. 14, no. 18, 2021.
5. A. Shkoruta, S. Mishra, and S. J. Rock, "Real-time image-based feedback control of laser powder bed fusion," *ASME Letters in Dynamic Systems and Control*, pp. 1–7, 2022.
6. T. Phillips, T. Ricker, S. Fish, and J. Beaman, "Design of a laser control system with continuously variable power and its application in additive manufacturing," *Additive Manufacturing*, vol. 34, 2020, doi: https://doi.org/10.1016/j.addma.2020.101173.

7. T. S. Ricker, "Position based Laser Power (Pblp) control for Selective Laser Sintering (Sls)," M.S. thesis, Department of Mechanical Engineering, University of Texas at Austin, Austin, TX, US, 2017.
8. H. Yeung, J. Neira, B. Lane, J. Fox, & F. Lopez, “Laser path planning and power control strategies for powder bed fusion systems,” International Solid Freeform Fabrication Symposium, Austin, TX, US, 2016.
9. G. Mohr, S. Altenburg, A. Ulbricht, P. Heinrich, D. Baum, C. Maierhofer, & K. Hilgenberg, “In-situ defect detection in laser powder bed fusion by using thermography and optical tomography—comparison to computed tomography,” *Metals*, vol. 10, no. 1, 2020, doi: https://doi.org/10.3390/met10010103.
10. P. A. Hooper, "Melt pool temperature and cooling rates in laser powder bed fusion," *Additive Manufacturing*, vol. 22, pp. 548–559, 2018.
11. S. Lee, J. Peng, D. Shin and Y. S. Choi, "Data Analytics approach for melt-pool geometries in metal additive manufacturing," *Science and Technology of Advanced Materials*, vol. 20, no.1, pp. 972–978, 2019.
12. H. Yeung, Z. Yang, & L. Yan, “A meltpool prediction based scan strategy for powder bed fusion additive manufacturing,” *Additive Manufacturing*, vol. 35, 2020, doi: https://doi.org/10.1016/j.addma.2020.101383.
13. I. Zhirnov, I. Yadroitsev, B. Lane, S. Mekhontsev, S. Grantham and I. Yadroitsava, "Influence of optical system operation on stability of single tracks in selective laser melting," US Department of Commerce, National Institute of Standards and Technology Advanced Manufacturing Series, 2019.

14. S. Sendino, M. Gardon, F. Lartategui, S. Martinez and A. Lamikiz, "The effect of the laser incidence angle in the surface of L-PBF processed parts," *Coatings*, vol. 10, no. 11, 2021.
15. K. Godineau, S. Lavernhe, & C. Tournier, "Calibration of galvanometric scan heads for additive manufacturing with machine assembly defects consideration," *Additive Manufacturing*, vol. 26, pp. 250–257, 2019, doi: https://doi.org/10.1016/j.addma.2019.02.003
16. V. Duma, P. Tankam, J. Huang, J. Won, & J. Rolland, "Optimization of galvanometer scanning for optical coherence tomography," *Applied Optics*, vol. 54, no. 17, pp. 5495-5507, 2015, doi: https://doi.org/10.1364/ao.54.005495
17. H. Yeung, J. Chen, G. Yang, Y. Guo, D. Lin, W. Tan, & J. Weaver, "Effect of spiral scan strategy on microstructure for additively manufactured stainless steel 17–4," *Manufacturing Letters*, vol. 29, pp. 1-4, 2021.
18. H. Yeung, B. Lane, J. Fox, F. Kim, J. Heigel, & J. Neira, "Continuous laser scan strategy for faster build speeds in laser powder bed fusion system," International Solid Freeform Fabrication Symposium, Austin, TX, US, 2017.
19. H. Zheng, Y. Wang, Y. Xie, S. Yang, R. Hou, Y. Ge, L. Lang, S. Gong, & H. Li, "Observation of vapor plume behavior and process stability at single-track and multi-track levels in laser powder bed fusion regime," *Metals*, vol. 11, no. 6, 2021, doi: https://doi.org/10.3390/met11060937
20. D. Deisenroth, S. Mekhontsev, B. Lane, L. Hanssen, I. Zhirnov, V. Khromchenko, S. Grantham, D. Cardenas-Garcia, & A. Donmez, "Measurement uncertainty of surface temperature distributions for laser powder bed fusion processes," Journal of Research of

the National Institute of Standards and Technology, vol. 126, 2021, doi: https://doi.org/10.6028/jres.126.013

21. C. Zhao, K. Fezzaa, R. Cunningham, H. Wen, F. De Carlo, L. Chen, A. Rollett, & T. Sun, "Real-time monitoring of laser powder bed fusion process using high-speed X-ray imaging and diffraction," *Scientific Reports*, vol. 7, no. 1, 2017, doi: https://doi.org/10.1038/s41598-017-03761-2
22. Y. Lu, Z. Yang, J. Kim, H. Cho, & H. Yeung, "Camera-based coaxial melt pool monitoring data registration for laser powder bed fusion additive manufacturing," *Advanced Manufacturing*, 2020, doi: https://doi.org/10.1115/imece2020-24546
23. S. Khairallah, A. Anderson, A. Rubenchik, & W. King, "Laser powder-bed fusion additive manufacturing: Physics of complex melt flow and formation mechanisms of pores, spatter, and denudation zones," *Acta Materialia*, vol. 108, pp. 36-45, 2016.
24. S. Khairallah, A. Martin, J. Lee, G. Guss, N. Calta, J. Hammons, & W. King, "Controlling interdependent meso-nanosecond dynamics and defect generation in metal 3D printing," *Science*, vol. 368, no. 6491, pp. 660-665, 2020.
25. A. Martin, N. Calta, S. Khairallah, J. Wang, P. Depond, A. Fong, & M. Matthews, "Dynamics of pore formation during laser powder bed fusion additive manufacturing," *Nature communications*, vol. 10, no. 1, 2019.
26. H. Yeung, B. Lane, M. Donmez, J. Fox, & J. Neira, "Implementation of advanced laser control strategies for powder bed fusion systems," *Procedia Manufacturing*, vol. 26, pp. 871-879, 2018.
27. H. Yeung, & B. Lane, "A residual heat compensation based scan strategy for powder bed fusion additive manufacturing," *Manufacturing letters*, vol. 25, pp. 56-59, 2020.

28. H. Yeung, B. Lane, & J. Fox, “Part geometry and conduction-based laser power control for powder bed fusion additive manufacturing,” *Additive manufacturing*, vol. 30, 2019.
29. O. Kwon, H. Kim, W. Kim, G. Kim, & K. Kim, “A convolutional neural network for prediction of laser power using melt-pool images in laser powder bed fusion,” *IEEE Access*, vol. 8, pp. 23255–23263, 2020, doi: https://doi.org/10.1109/access.2020.2970026
30. Z. Yang, Y. Lu, H. Yeung, & S. Krishnamurty, “Investigation of Deep Learning for real-time melt pool classification in additive manufacturing,” IEEE 15th International Conference on Automation Science and Engineering (CASE), Vancouver, BC, Canada, pp. 640–647, 2019, doi: https://doi.org/10.1109/coase.2019.8843291
31. F. Ogoke, & A. Farimani, “Thermal control of laser powder bed fusion using deep reinforcement learning,” *Additive Manufacturing*, vol. 46, 2021, doi: https://doi.org/10.1016/j.addma.2021.102033
32. B. Lane, & H. Yeung, “Process monitoring dataset from the Additive Manufacturing Metrology testbed (AMMT): Overhang part X4,” Journal of Research of the National Institute of Standards and Technology, vol. 125, 2020 doi: https://doi.org/10.6028/jres.125.027
33. K. Ettaieb, K. Godineau, S. Lavernhe, & C. Tournier, “Offline laser power modulation in LPBF additive manufacturing including kinematic and technological constraints,” *Rapid Prototyping Journal*, vol. 29, no. 1, pp. 80-91, 2020, doi: https://doi.org/10.1108/RPJ-02-2022-0062.
34. K. C. Mills, *Recommended values of thermophysical properties for selected commercial alloys*, Woodhead Publishing, 2002.

35. Scanlab GmbH, *Installation and Operation RTC5 PCI Board, RTC5 PCIe Board, RTC5 PC/104-Plus Board and RTC5 PCIe/104 Board for Real Time Control of Scan Heads and Lasers Rev.1.14.3*. Puchheim, Germany: Scanlab GMBH, 2021.
36. Raylase GmbH, *SP-ICE-3 User's Manual, Rev. v1.39.1*. Weßling, Germany: Raylase GmbH, 2017.
37. A. GmbH, "Common Layer Interface (CLI)," Common layer interface (CLI), https://aconity3d.atlassian.net/servicedesk/customer/portal/2/article/77561885?src=-1856702863 (accessed Sep. 15, 2023).
38. A. GmbH, "CLI+ / CLI Plus," Aconity3D Help Center, https://aconity3d.atlassian.net/servicedesk/customer/portal/2/article/199458870?src=1103665299 (accessed Sep. 15, 2023).
39. A. Bernal et al., "Flat Field Correction," Drive AM - LPBF Content Conversion, https://driveam.atlassian.net/wiki/spaces/TCC/pages/81363995/Flat+Field+Correction (accessed Sep. 15, 2023).
40. J. Aguilar and H. Taylor, "Sigma data MatLab Monitoring data process," Drive AM - LPBF Monitoring, https://driveam.atlassian.net/wiki/spaces/LPBFMon/pages/90210476/Sigma+data+MatLab+Monitoring+data+process (accessed Sep. 15, 2023).

APPENDIX

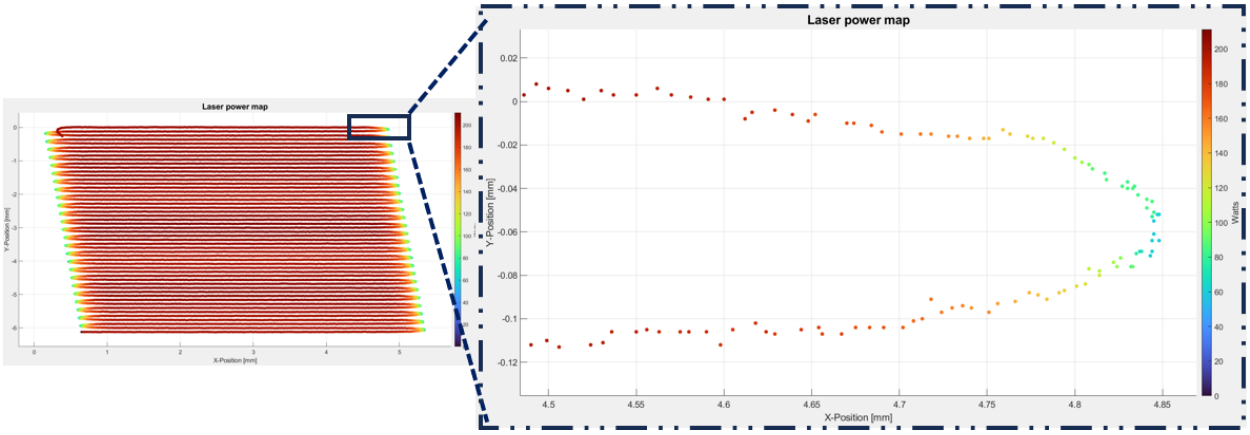

Figure 63: Power map for VSV1 RP.

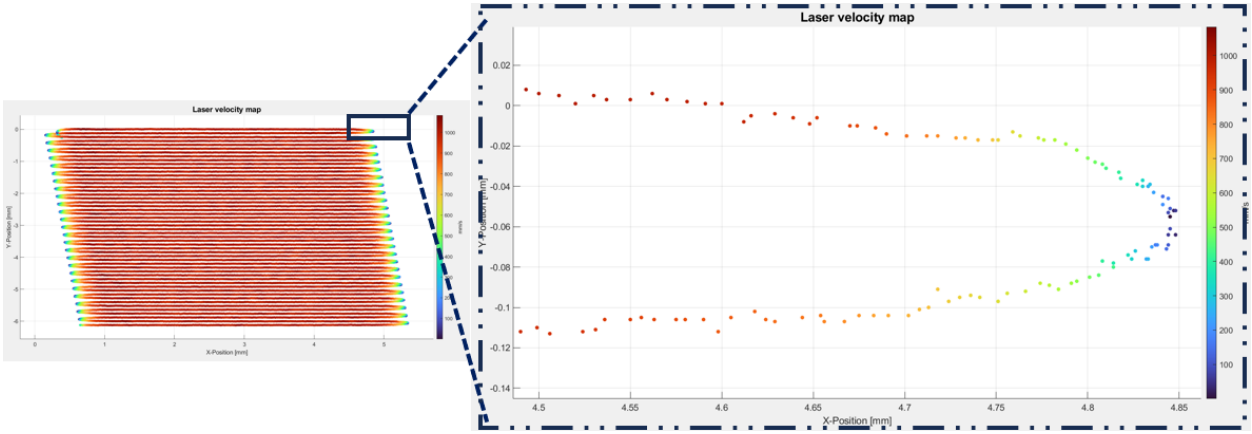

Figure 64: Velocity map for VSV1 RP.

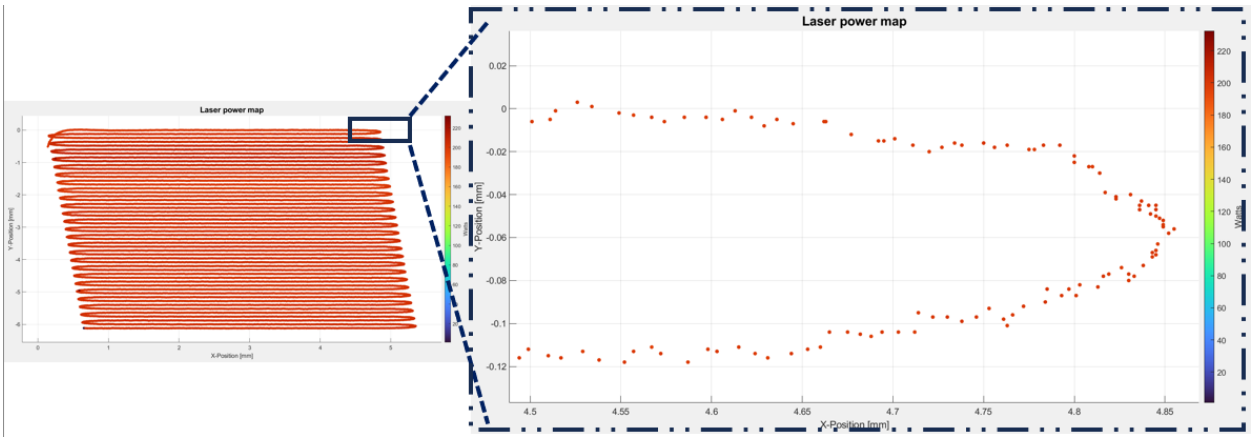

Figure 65: Power map for VSV1 CP.

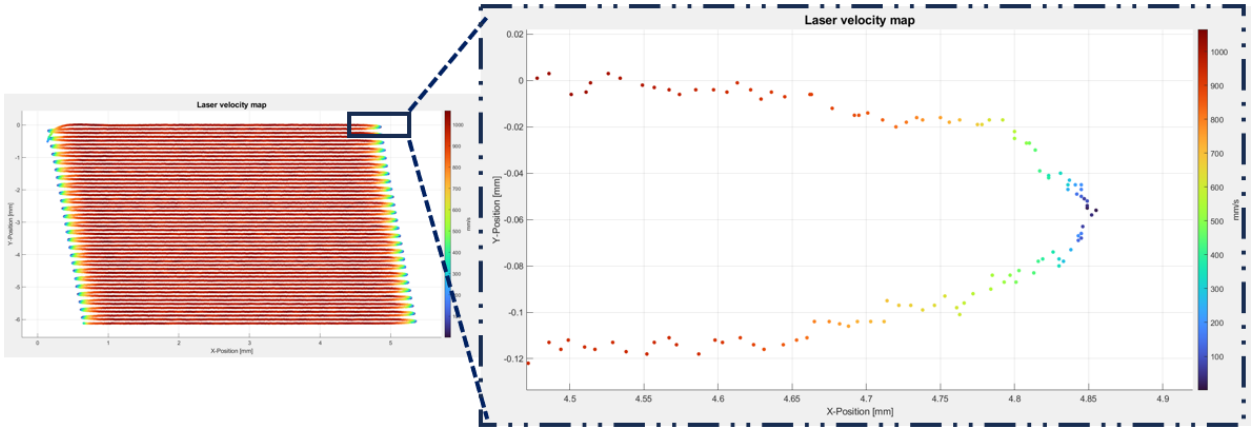

Figure 66: Velocity map for VSV1 CP.

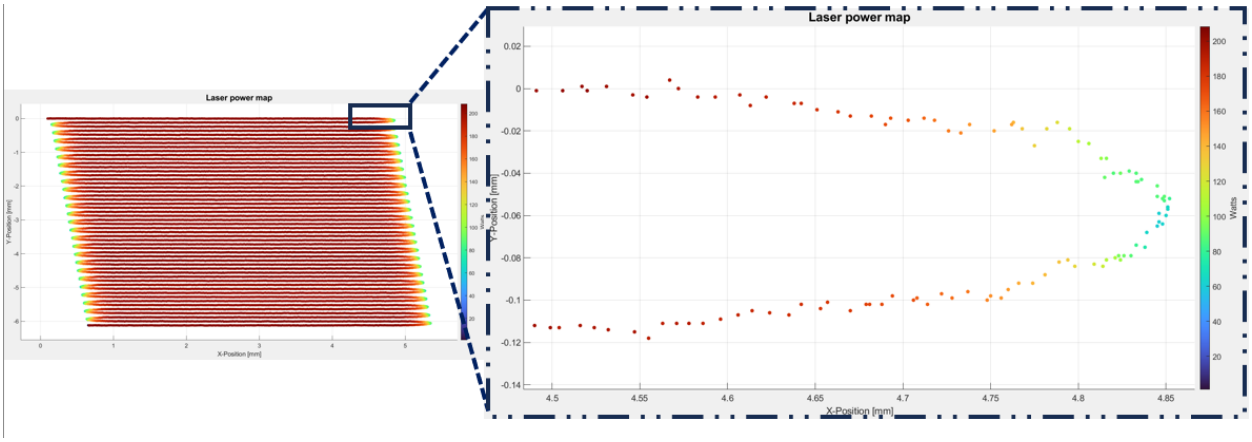

Figure 67: Power map for VSV2 RP.

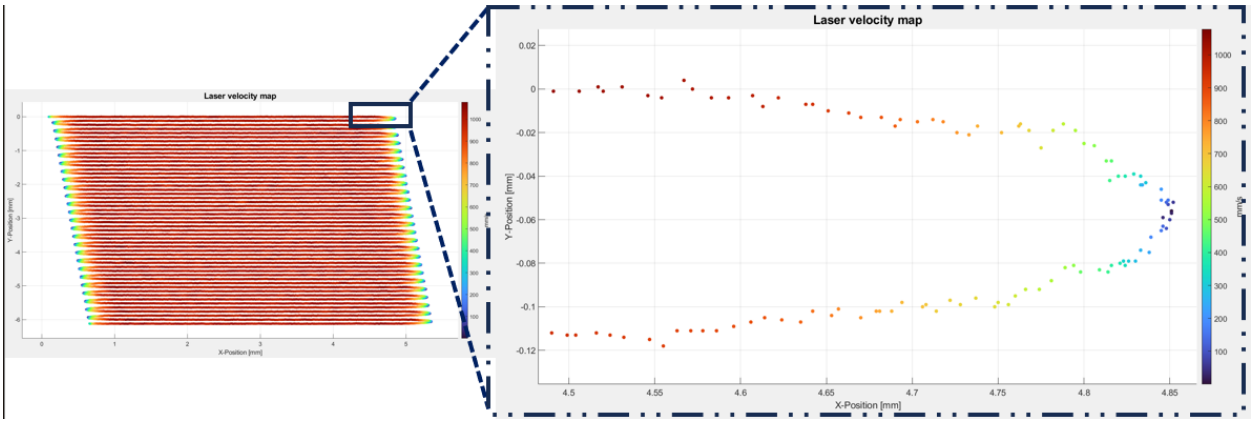

Figure 68: Velocity map for VSV2 RP.

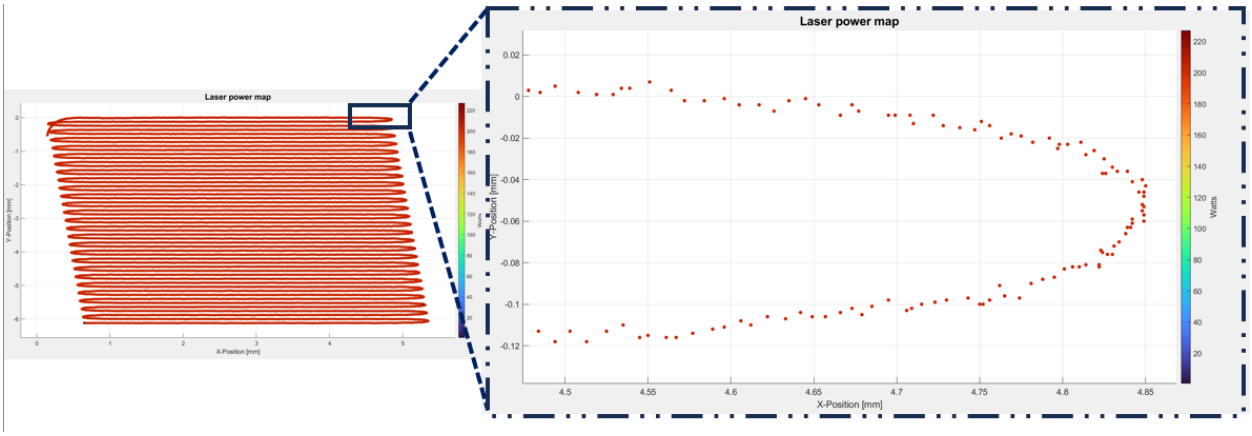

Figure 69: Power map for VSV2 CP.

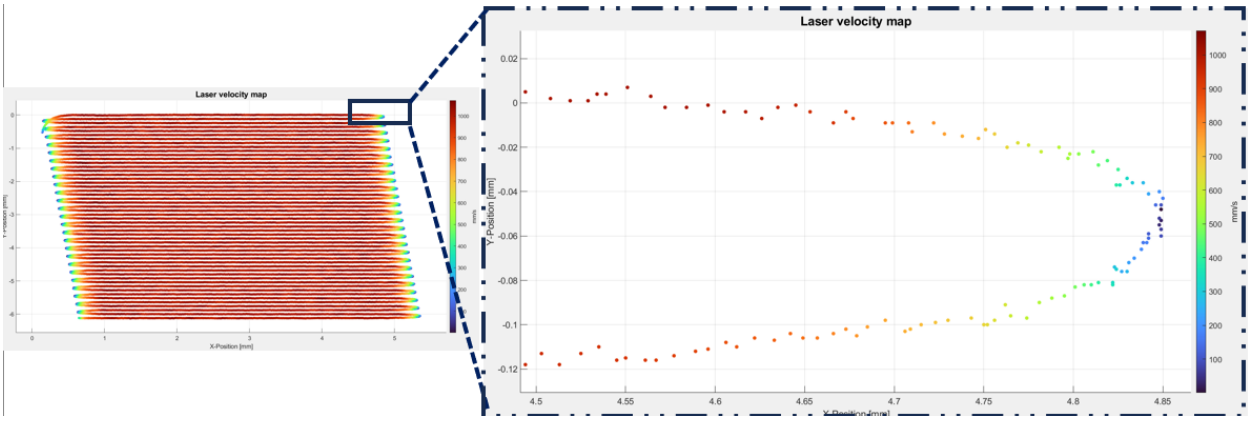

Figure 70: Velocity map for VSV2 RP.

Figure 71: Surface image of the RP scan for VSV1.

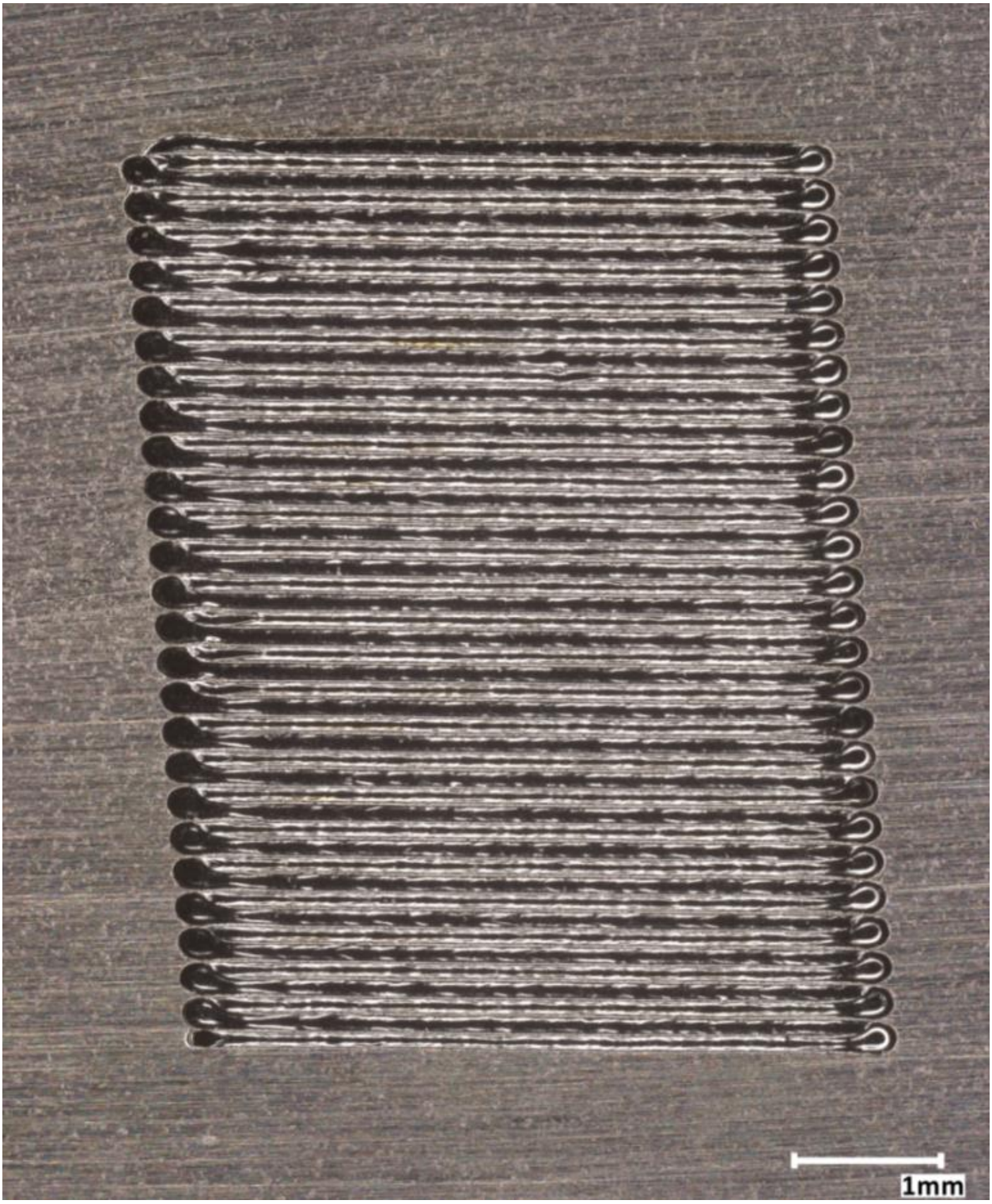

Figure 72: Surface image of the CP scan for VSV1.

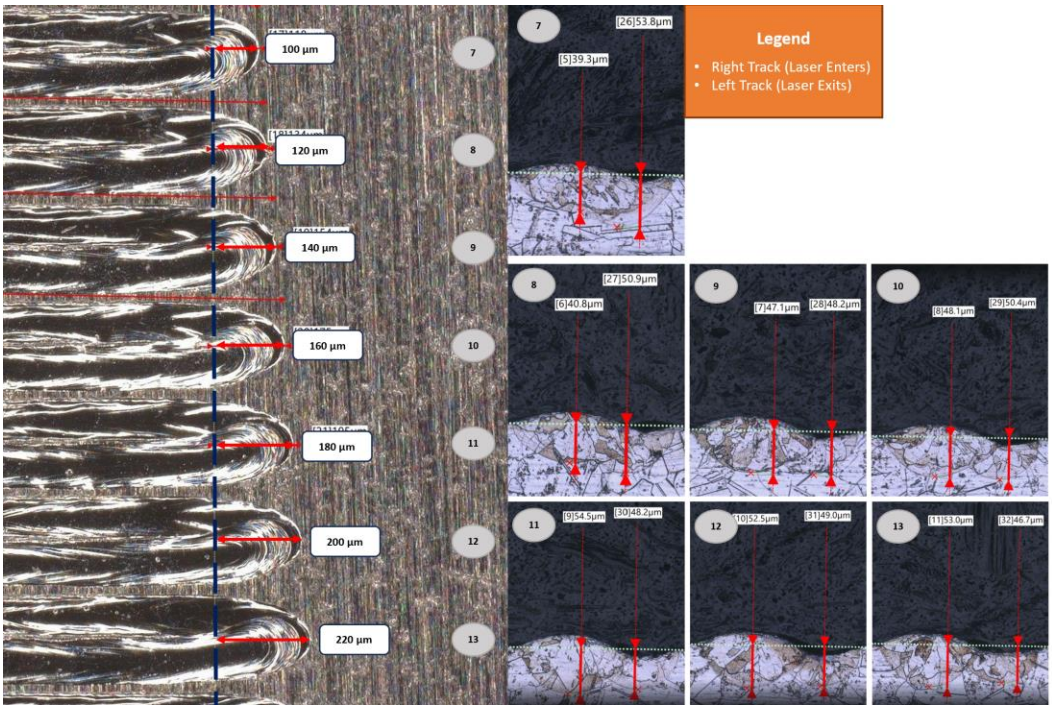

Figure 73: RP sections 7-13 based on surface distance VSV1.

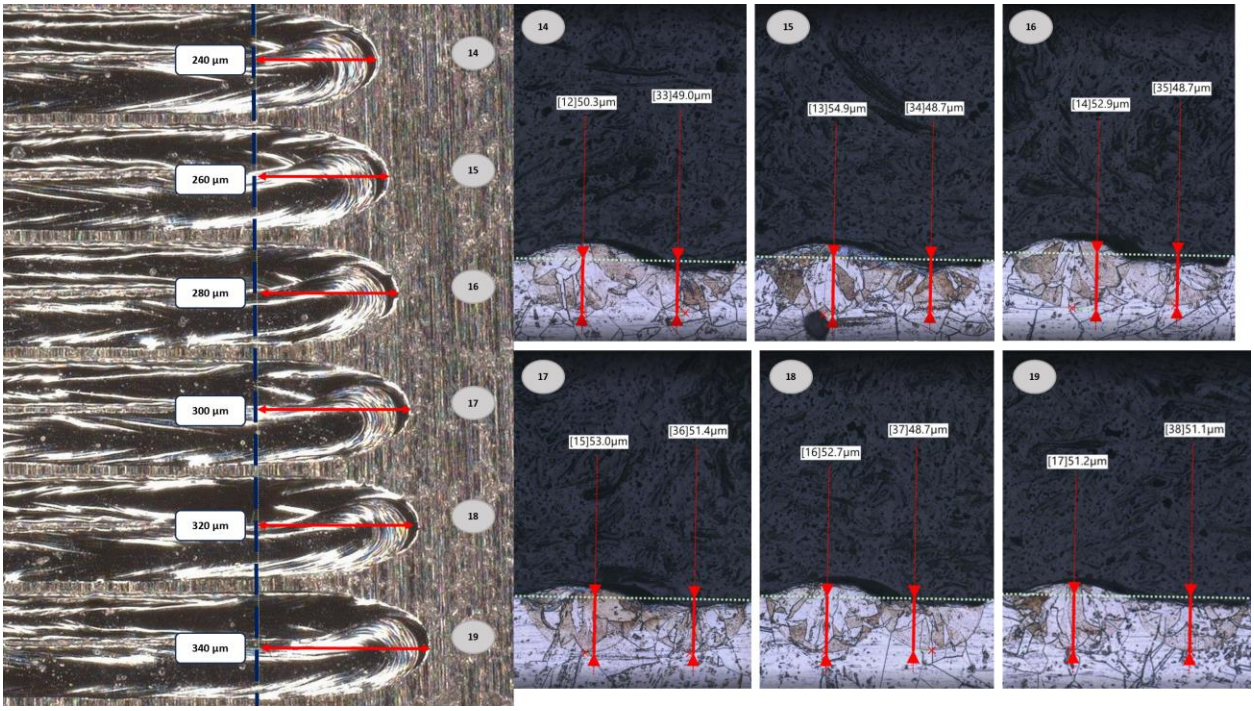

Figure 74: RP sections 14-19 based on surface distance VSV1.

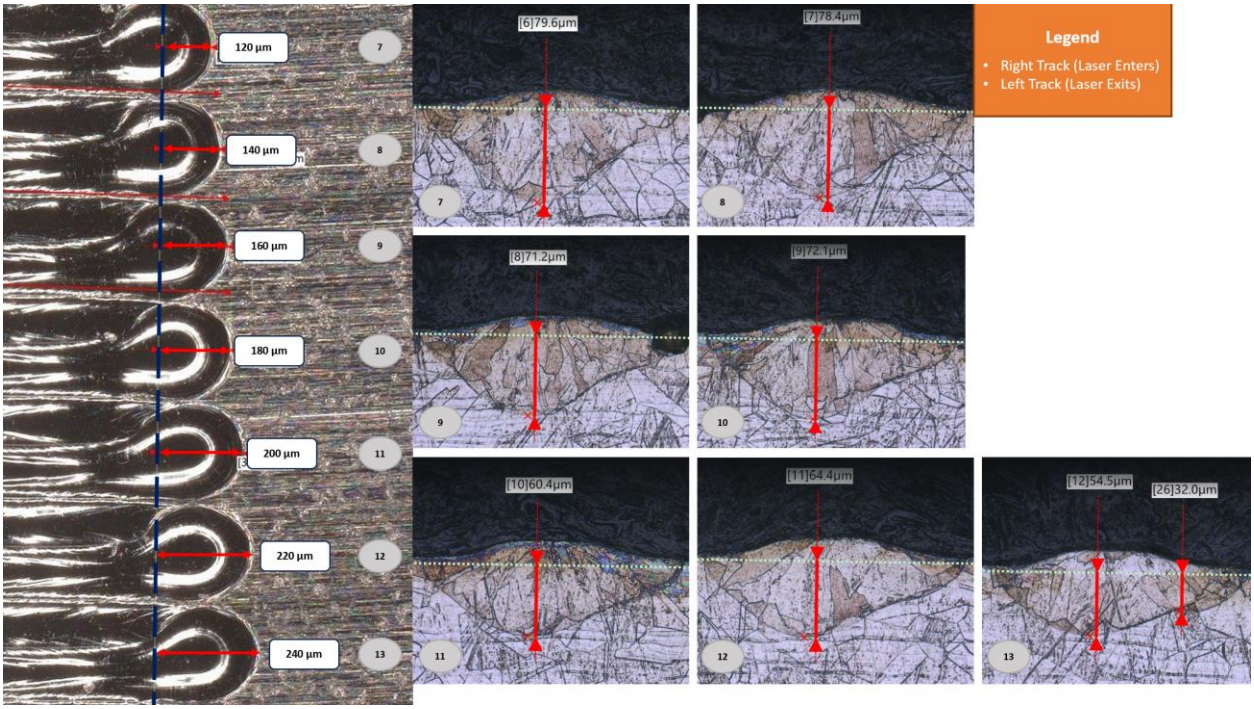

Figure 75: CP sections 7-13 based on surface distance VSV1.

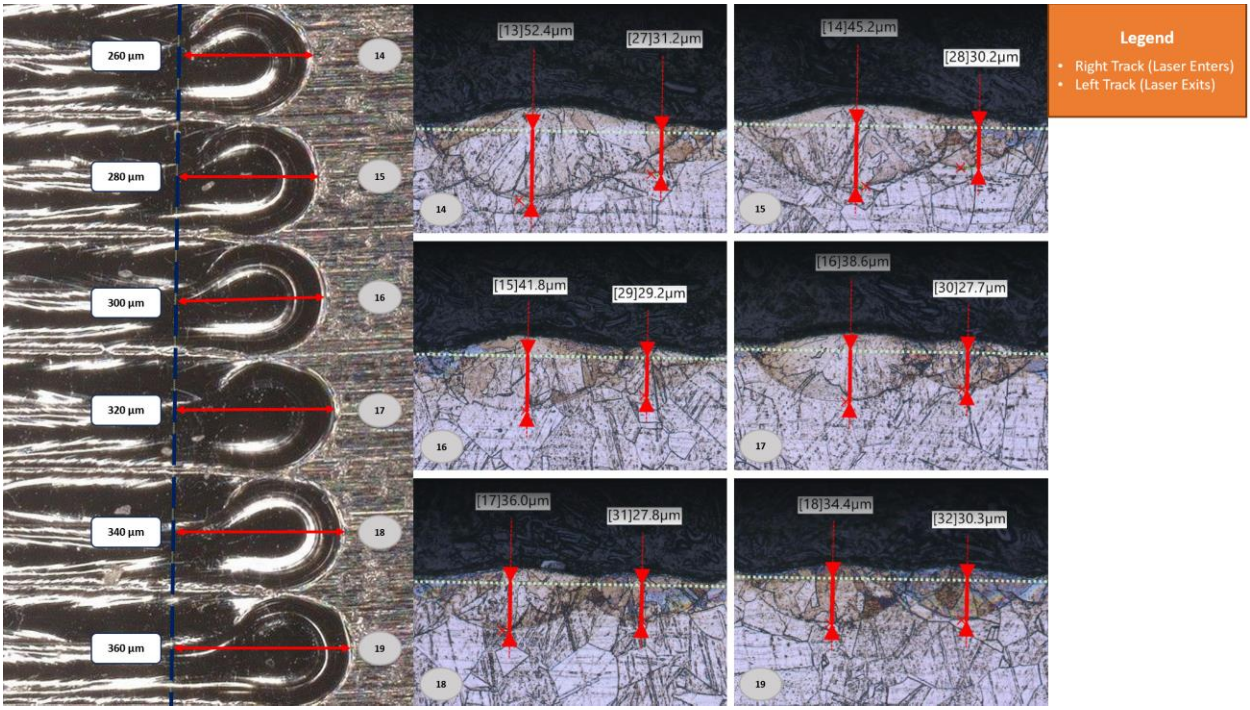

Figure 76: CP sections 14-19 based on surface distance VSV1.

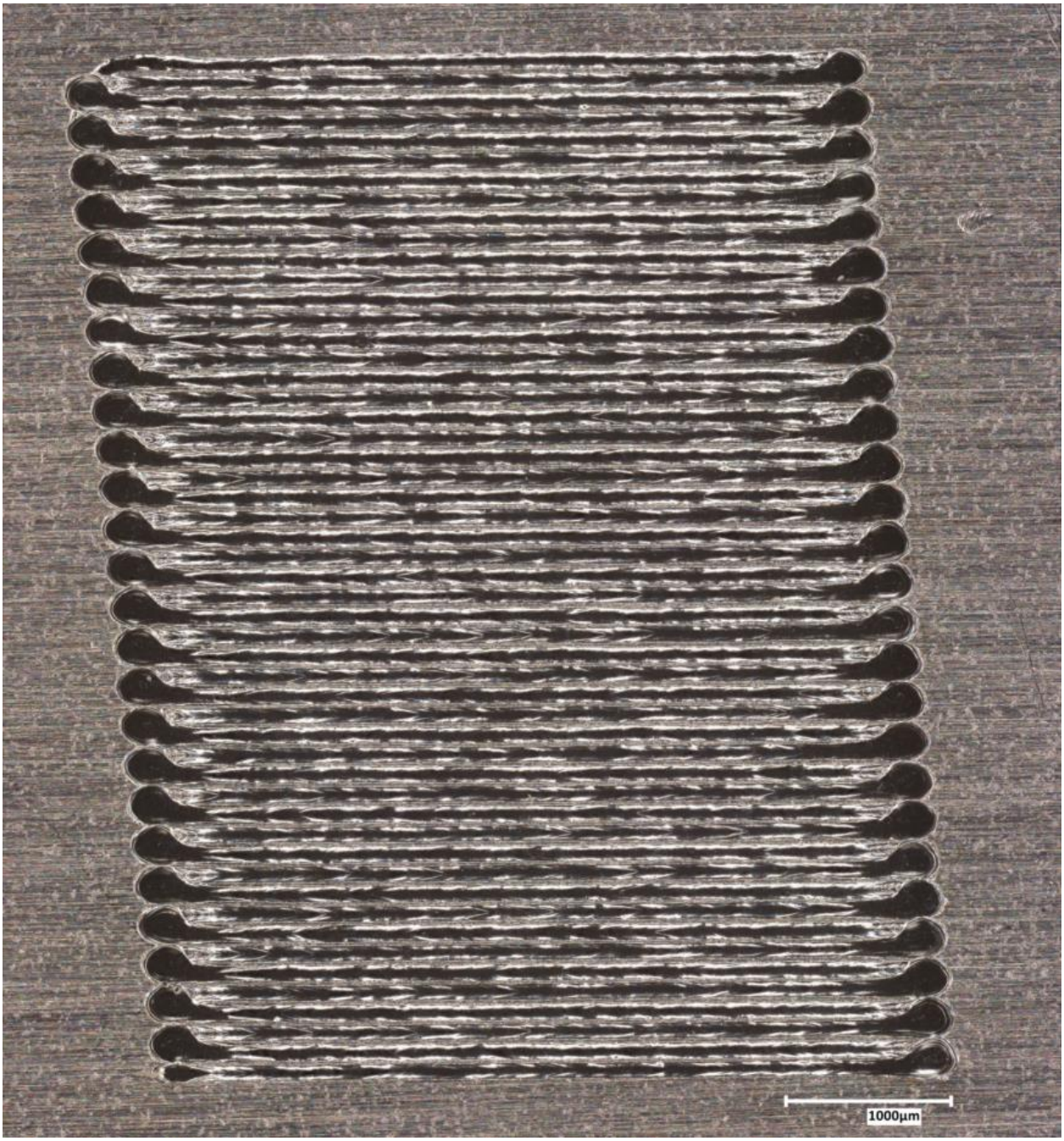

Figure 77: Surface image of the CP scan for VSV2.

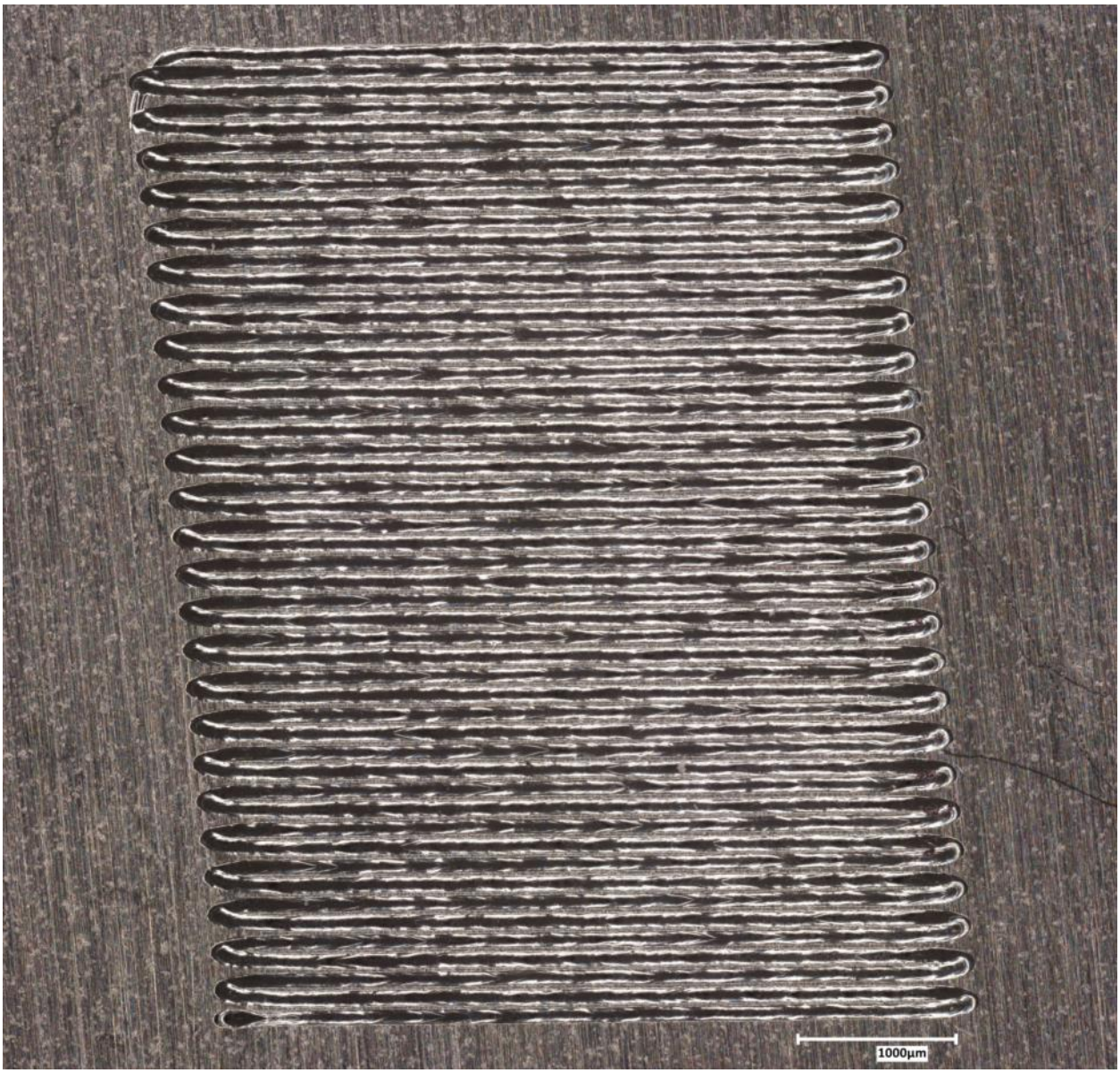

Figure 78: Surface image of the RP scan for VSV2.

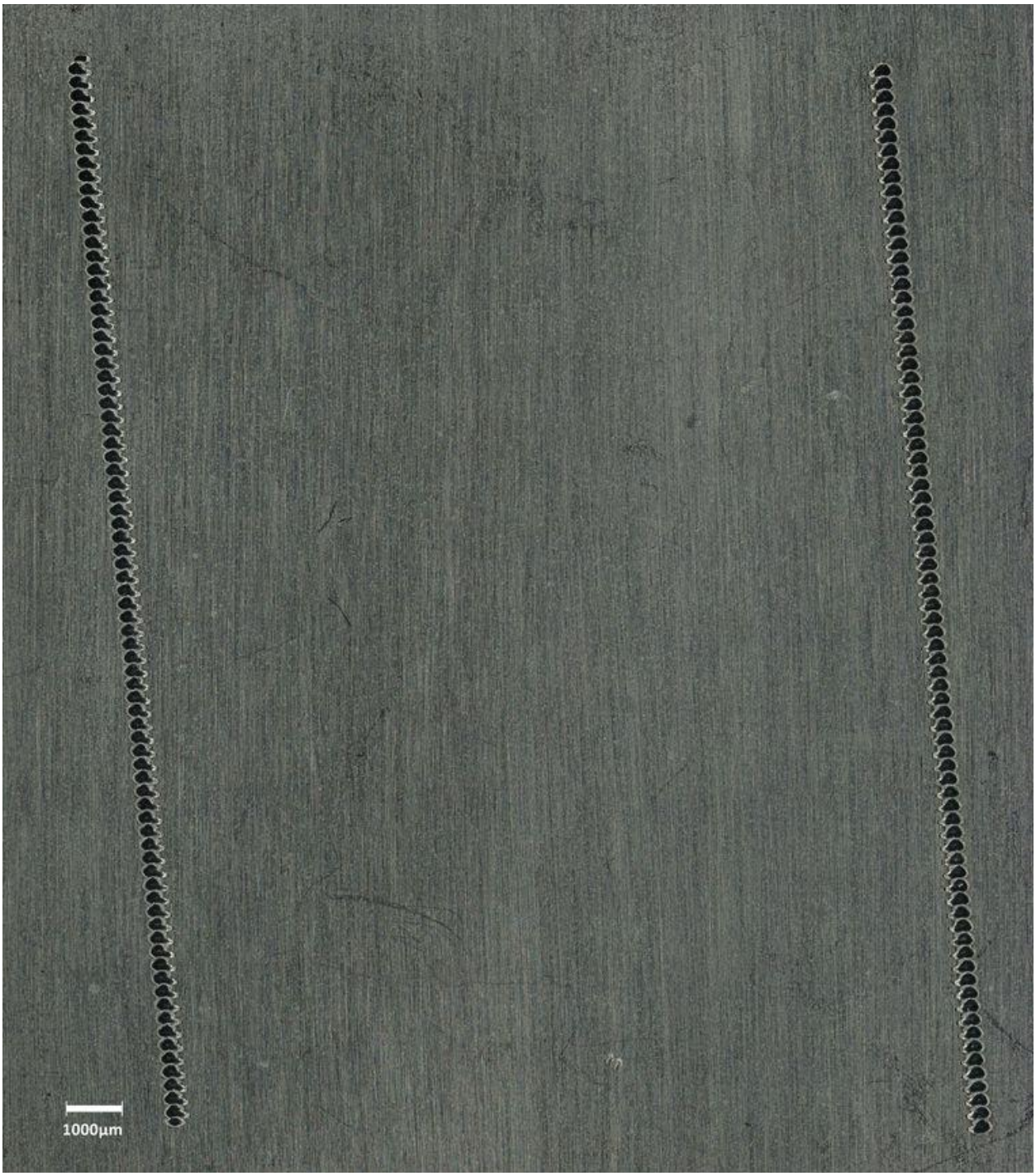

Figure 79: Surface image of the ramp index scan for VSV3.

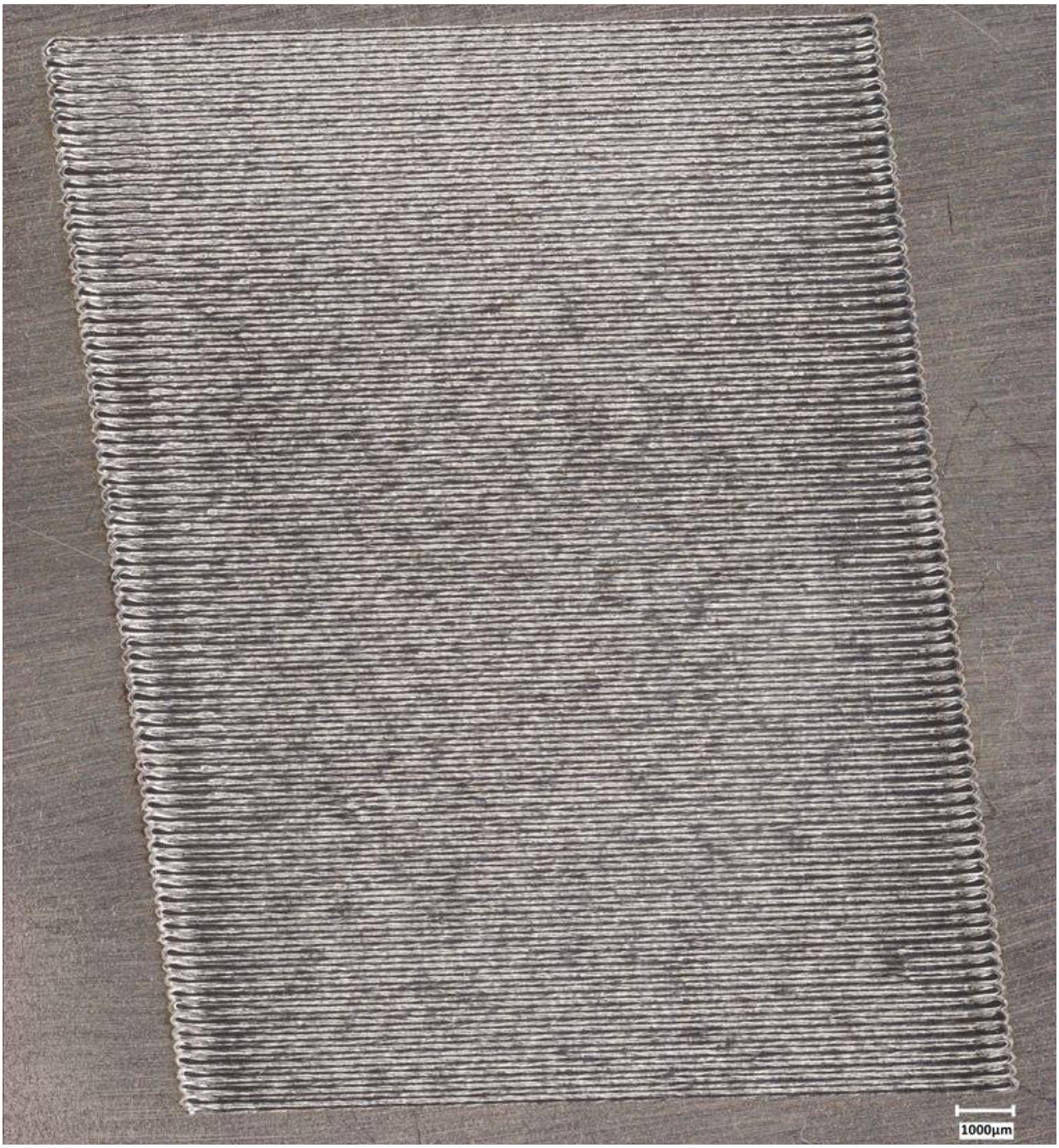

Figure 80: Surface image of the CP scan for VSV3.

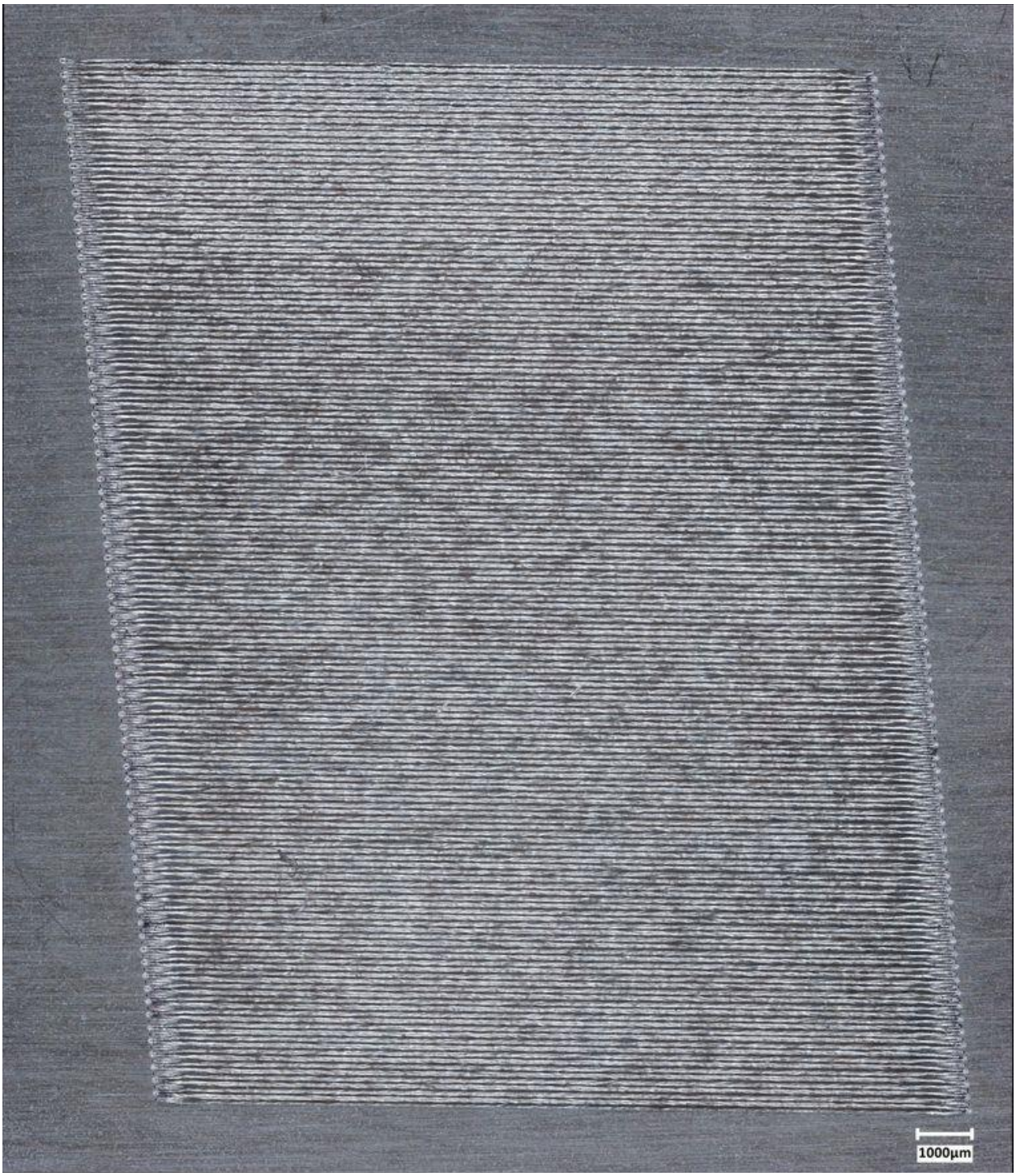

Figure 81: Surface image of the CV scan for VSV3.

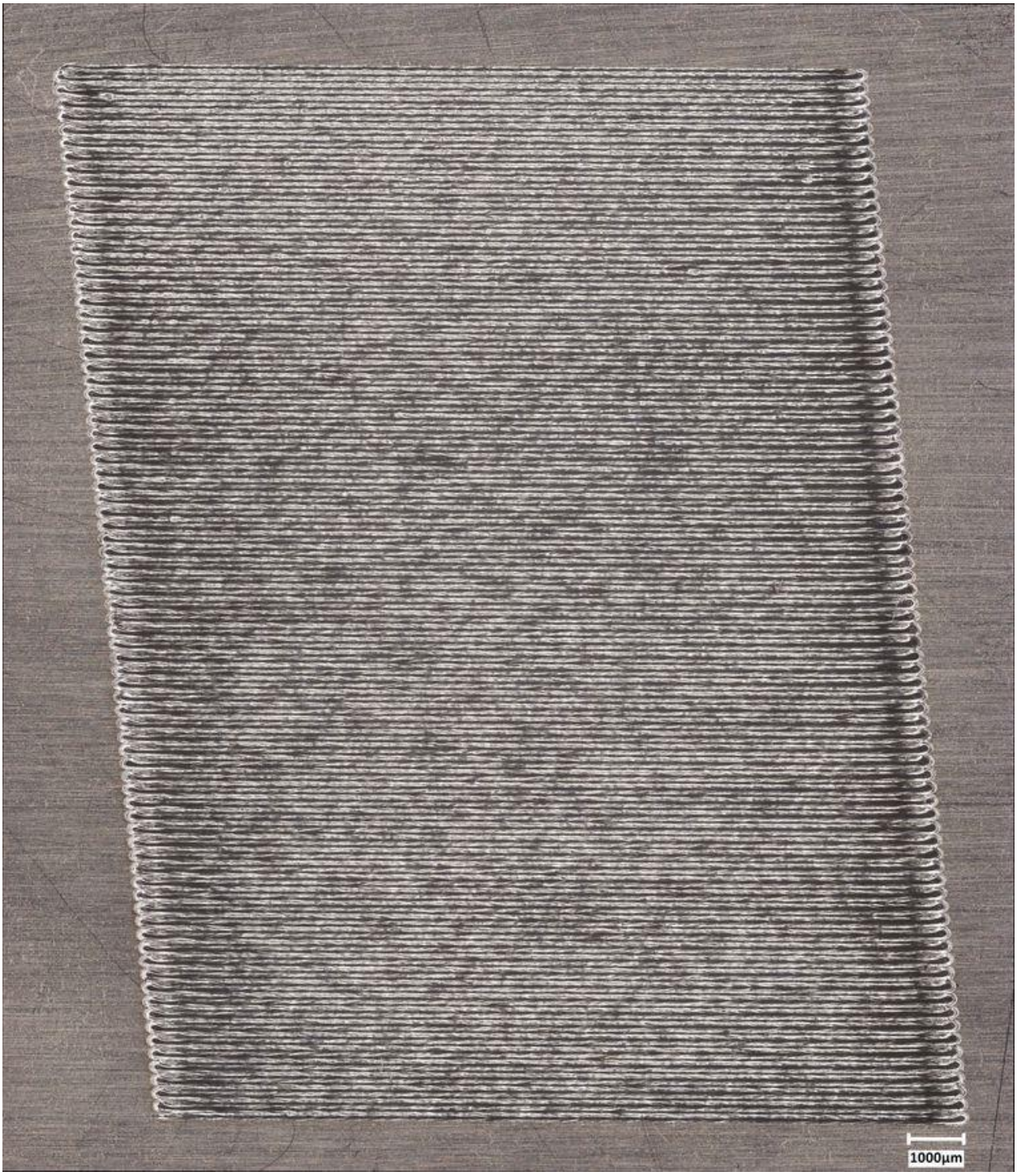

Figure 82: Surface image of the RP scan for VSV3.

Figure 83: Surface image of the SW scan for VSV3.

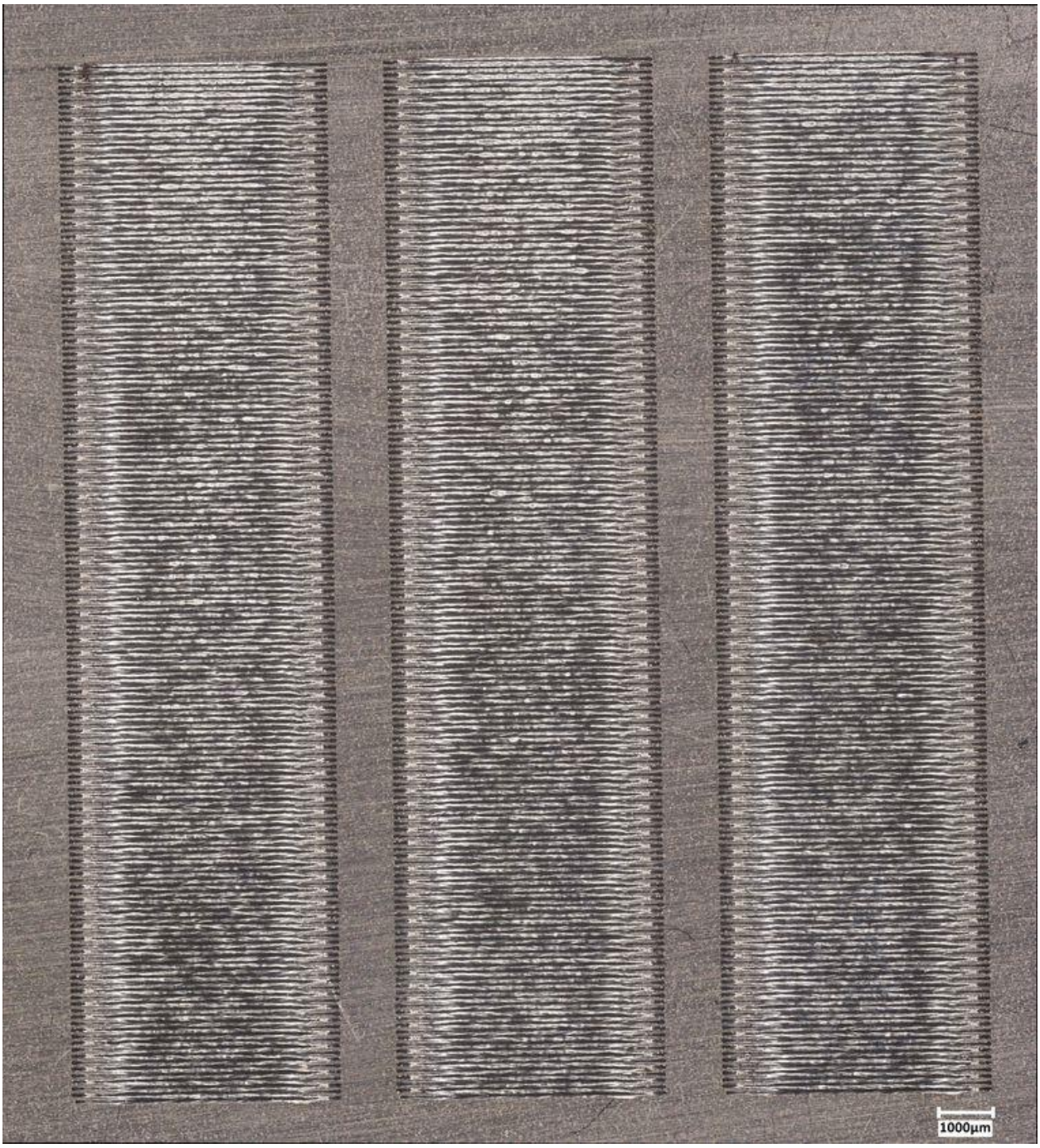

Figure 84: Surface image of the RP type 2 parallel scan for VSV3.

Figure 85: Surface image of the RP type 2 staggered scan for VSV3.

Velocity Approximation Script

```
%% script setup


clear
close all
clc

% elapsed script time
tic

% import parameters
delimiterIn = (' ');
headerlinesIn = 27;
file = 'corrected 23-09-22-Type1.pcd';
% file = '23-08-30-job3-nosw-0.05.pcd';
format long

% file specification and parsing
a = importdata(file, delimiterIn, headerlinesIn);


%% variable initialization


% filter out excess data
volt = a.data(:, 8);
volt = circshift(volt, 8); %#ok<NASGU>
s = a.data(:, 9);
s = circshift(s, 8);
state_clip = find(s >= 1);
state_start = state_clip(1);
state_end = state_clip(end);
a.data = a.data((state_start):(state_end - 8), :);

% calibration factors
% kx = 600 / (2^20 - 1);
% ky = 600 / (2^20 - 1);

% inverted correction factors
% note pcd axes are swapped due to gloal and local co-ord differences
xoffset = 40.5;
yoffset = 82;
zoffset = -4;
xscaling = 1;
yscaling =  1;
zscaling = 1;

% data seperation
t = a.data(:, 1);
x = (a.data(:, 2)) / xscaling - xoffset;
y = (a.data(:, 3)) / yscaling + yoffset;
z = (a.data(:, 4)) / zscaling + zoffset;
```

```
pyro1 = a.data(:, 5);
pyro2 = a.data(:, 6);
volt = a.data(:, 8);
c = 0.0189;
b = 0.0379;
p = ((volt / 0.4) - b) ./ (1000 * c);
s = a.data(:, 9);

% create rotation matrix
theta = 90; % to rotate 90 counterclockwise
R = [cosd(theta) -sind(theta); sind(theta) cosd(theta)];
% rotate your point(s)
coords = [x y]'; % arbitrarily selected
rotcoords = R * coords;
x = (rotcoords(1, :))';
y = (rotcoords(2, :))';

state = [y, x, s];
zero = state(:, 3) == 0;
one = state(:, 3) == 1;
off = state;
off(one, :) = [];
on = state;
on(zero, :) = [];

% constants
timesteps = 0.00001;
fs = 1e5;
T = t * timesteps * 0.1;

% x velocity and acceleration
vx = diff(x) / timesteps;
Vx = [0; vx];
ax = diff(Vx(:, 1)) / timesteps;
Ax = [0; ax];

% y velocity and acceleration
vy = diff(y) / timesteps;
Vy = [0; vy];
% ay = diff(Vy(:, 1)) / timesteps;
% Ay = [0; ay];

% total velocity and acceleration
vtotal = sqrt((Vx.^2) + (Vy.^2));
vtotal = vtotal(30:end, :); % prevent midcross jump effect
veltt = array2timetable(vtotal, 'SampleRate', fs);
powtt = array2timetable(p, 'SampleRate', fs);

% timetable conversion
% veltty = array2timetable(Vy, 'SampleRate', fs);
velttx = array2timetable(Vx, 'SampleRate', fs);
accelttx = array2timetable(Ax, 'SampleRate', fs);
% acceltty = array2timetable(Ay, 'SampleRate', fs);
```

```
%% data filtering


% % unfiltered velocity over time
% figure
% scatter(T, vtotal, 'filled', 'DisplayName', 'Raw Vtotal')
% xlabel('Time [s]')
% ylabel('Velocity [mm/s]')
% title('Unfiltered total velocity over time')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % unfiltered X-velocity over time
% figure
% scatter(T, Vx, 'filled', 'DisplayName', 'Raw Vx')
% xlabel('Time [s]')
% ylabel('Velocity [mm/s]')
% title('Unfiltered Xvelocity over time')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % unfiltered velocity power spectrum
% figure
% [pp,f] = pspectrum(veltt);
% scatter(f,pow2db(pp))
% xlabel('Frequency [Hz]')
% ylabel('Power Spectrum [dB]')
% title('Unfiltered Vtotal frequency domain')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % unfiltered velocity power spectrum
% figure
% [pp,f] = pspectrum(velttx);
% scatter(f,pow2db(pp))
% xlabel('Frequency [Hz]')
% ylabel('Power Spectrum [dB]')
% title('Unfiltered Vx frequency domain')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % unfiltered velocity spectrogram
% figure
% pspectrum(veltt, "spectrogram")

% f0vx = lowpass(velttx, 61);
% % f0vy = lowpass(veltty, 61);
fveltt = lowpass(veltt, 49, Steepness = 0.85);
fvelttx = lowpass(velttx, 61, Steepness = 0.85);
% fveltt = bandpass(veltt, [35225 35593]);
f0ax = bandpass(accelttx, [49 1697], Steepness = 0.85);
```

```
% unfiltered velocity power spectrum
% figure
% [pp,f] = pspectrum(powtt);
% scatter(f,pow2db(pp))
% xlabel('Frequency [Hz]')
% ylabel('Power Spectrum [dB]')
% title('Unfiltered power frequency domain')
% xlim padded
% ylim padded
% legend('Location', 'best')

% fpowtt = lowpass(powtt, 49, Steepness = 0.95);
% % filtered velocity over time & power spectrum
% figure
% scatter(T, powtt.p, 'filled', 'DisplayName', 'Raw Power')
% hold on
% scatter(T, fpowtt.p, 'filled', 'DisplayName', 'Filtered Power')
% xlabel('Time [s]')
% ylabel('Power [W]')
% title('Raw vs Filtered Power over time')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % total velocity
% % speedlim = 9e3;
% vtotal = sqrt((f0vx.Vx.^2) + (f0vy.Vy.^2));
% % vtotal(vtotal >= speedlim) = nan;

% % filtered velocity over time & power spectrum
% figure
% subplot(2, 1, 1)
% scatter(T, veltt.vtotal, 'filled', 'DisplayName', 'Raw Vtotal')
% hold on
% scatter(T, fveltt.vtotal, 'filled', 'DisplayName', 'Filtered Vtotal')
% xlabel('Time [s]')
% ylabel('Velocity [mm/s]')
% title('Raw vs Filtered Vtotal over time')
% xlim padded
% ylim padded
% legend('Location', 'best')
% subplot(2, 1, 2)
% [p1,f1] = pspectrum(veltt);
% [p2,f2] = pspectrum(fveltt);
% scatter(f1, pow2db(p1), 'filled', 'DisplayName', 'Raw Vtotal')
% hold on
% scatter(f2, pow2db(p2), 'filled', 'DisplayName', 'Filtered Vtotal')
% xlabel('Frequency [Hz]')
% ylabel('Power Spectrum [dB]')
% title('Raw vs Filtered Vtotal power spectrum')
% xlim padded
% ylim padded
% legend('Location', 'best')
```

```
% % laser velocity map
% figure
% % scatter3(x, y, fveltt.vtotal, [], fveltt.vtotal, 'filled', 'DisplayName',
'Velocity');
% scatter3(x, y, fvelttx.Vx, [], abs(fvelttx.Vx), 'filled', 'DisplayName', 'X-
Velocity');
% xlabel('X-Position [mm]')
% ylabel('Y-Position [mm]')
% zlabel('Velocity [mm/s]')
% view(0, 90)
% colormap(turbo)
% hc = colorbar;
% ylabel(hc,'mm/s', 'FontSize', 16);
% title('Laser velocity map')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % filtered X-velocity over time
% figure
% scatter(T, abs(fvelttx.Vx), 'filled', 'DisplayName', 'Filtered Vx')
% xlabel('Time [s]')
% ylabel('Velocity [mm/s]')
% title('Filtered X-Velocity over time')
% xlim padded
% ylim padded
% legend('Location', 'best')
```

```
%% velocity bounds

[c, midlev] = midcross(abs(fvelttx.Vx), fs);
figure
midcross(abs(fvelttx.Vx), fs);
hold on
V = abs(fvelttx.Vx);
t = ((t - (t(1) + 0)) / 10) / fs;

[pks,locs] = findpeaks(V, t, 'MinPeakHeight', 800);
findpeaks(V, t, 'MinPeakHeight', 800)

xlineleft = [];
xlineright = [];
xlinedecelleft = [];
xlinedecelright = [];
xlineaccelleft = [];
xlineaccelright = [];
xlinedecel = [];
```

```
xlineaccel = [];

index = locs;
c = c(2:end-1, :);

for i = 1:length(c)
    if midlev > 500
        if mod(i,2) ~= 0
            lowindex = index;
            minindex = index < c(i);
            lowindex(minindex < 1) = [];
            lowindex = lowindex(end);
            xlineleft = [xlineleft; lowindex];
        else
            highindex = index;
            maxindex = index > c(i);
            highindex(maxindex < 1) = [];
            highindex = highindex(1);
            xlineright = [xlineright; highindex];
        end
    xlines = [xlineleft; xlineright];
    end
end
%
%     else
%         if mod(i,2) ~= 0
%             leftdecelindex = index;
%             minindex = index < c(i);
%             leftdecelindex(minindex < 1) = [];
%             leftdecelindex = leftdecelindex(end);
%
%             rightdecelindex = index;
%             maxindex = index > c(i);
%             rightdecelindex(minindex < 1) = [];
%             rightdecelindex = rightdecelindex(1);
%
%             xlinedecel = [xlinedecel; leftdecelindex; rightdecelindex];
%         else
%             leftaccelindex = index;
%             minindex = index < c(i);
%             leftaccelindex(minindex < 1) = [];
%             leftaccelindex = leftaccelindex(end);
%
%             rightaccelindex = index;
%             maxindex = index > c(i);
%             rightaccelindex(minindex < 1) = [];
%             rightaccelindex = rightaccelindex(1);
%
%             xlineaccel = [xlineaccel; leftaccelindex; rightaccelindex];
%         end
%     xlines = [xlinedecel; xlineaccel];
%
%     end
% end
%
```

```
ROI = ismember(t, xlines);
hold on
plot(t,V,t(ROI),V(ROI),'bsquare','LineWidth', 2)
xlabel('Time [s]')
ylabel('V_T_o_t_a_l [mm/s]')
title('Velocity Bound Identification')
xlim padded
ylim padded
legend('Location', 'best')


%% velocity approximation


% picosmaxa = picosmaxa(:, 1);
% decelstart = picosmaxa(1:2:end);
%
% picosmina = picosmina(:, 1);
% accelstart = picosmina(2:2:end);
%
% picosmaxb = picosmaxb(:, 1);
% accelend = picosmaxb(2:2:end);
%
% picosminb = picosminb(:, 1);
% decelend = picosminb(1:2:end);

accelstart = xlineleft;
accelend = xlineright;
accelmin = abs(accelstart - accelend);
accelmin = uint8(min(accelmin) * fs);

% number of vectors
numvectors = 199; % (length(picosmina)) / 2;

% region 1 timestep intervals
decelts = nan(500, numvectors);
accelts = nan(500, numvectors);

for i = 1:numvectors
    % resultdecelts = round(decelstart(i, :) * fs:decelend(i, :) * fs)';
    % decelts(1:numel(resultdecelts),i) = resultdecelts;
    resultaccelts = round(accelstart(i, :) * fs:accelend(i, :) * fs)';
    accelts(1:numel(resultaccelts),i) = resultaccelts;
end

% decelts = decelts(1:decelmin, :);
accelts = accelts(1:accelmin, :);
% decelexpvel = nan(decelmin, numvectors);
accelexpvel = nan(accelmin, numvectors);

for i = 1:numvectors
    % decelexpvel(:, i) = abs(fveltt.Vx(decelts(:, i), :));
    accelexpvel(:, i) = abs(fvelttx.Vx(accelts(:, i), :));
end
```

```
% decelexpvelavg = mean(decelexpvel, 2);
accelexpvelavg = mean(accelexpvel, 2);

% variable qp based on different bound lengths, extrapolation
% qp1 = double(1:decelmin)';
qp2 = double(1:accelmin)';

% region 1 time intervals
% decelt = decelts ./ fs;
accelt = accelts ./ fs;

% nth degree polynomial approximation of averaged velocity for accel/decel
% [P1, S1, Mu1] = polyfit(qp1, decelexpvelavg, 3);
[P2, S2, Mu2] = polyfit(qp2, accelexpvelavg, 4);

% calculation of points based on approximated profile for region 1
% ypoly1 = polyval(P1, qp1, S1, Mu1);
ypoly2 = polyval(P2, qp2, S2, Mu2);

% R^2 for polynomial regression of Mean Vx
% yresid1 = decelexpvelavg - ypoly1;
% SSresid1 = sum(yresid1.^2);
% SStotal1 = (length(decelexpvelavg)-1) * var(decelexpvelavg);
% rsqDecelMean = 1 - (SSresid1 / SStotal1)

yresid2 = accelexpvelavg - ypoly2;
SSresid2 = sum(yresid2.^2);
SStotal2 = (length(accelexpvelavg)-1) * var(accelexpvelavg);
rsqAccelMean = 1 - (SSresid2 / SStotal2)

% R^2 for polynomial regression of Individual Vx
% rsqDecel = nan(1, 99);
rsqAccel = nan(1, 199);

for i = 1:199
    % yresid1 = decelexpvel(:, i) - ypoly1;
    % SSresid1 = sum(yresid1.^2);
    % SStotal1 = (length(decelexpvel(:, i))-1) * var(decelexpvel(:, i));
    % rsqn1 = 1 - (SSresid1 / SStotal1);
    % rsqDecel(:, i) = rsqn1;

    yresid2 = accelexpvel(:, i) - ypoly2;
    SSresid2 = sum(yresid2.^2);
    SStotal2 = (length(accelexpvel(:, i))-1) * var(accelexpvel(:, i));
    rsqn2 = 1 - (SSresid2 / SStotal2);
    rsqAccel(:, i) = rsqn2;
end

% rsqDecelMin = min(rsqDecel)
% rsqDecelMax = max(rsqDecel)
% rsqDecelAvg = mean(rsqDecel)
% rsqDecelStd = std(rsqDecel)

rsqAccelMin = min(rsqAccel)
```

```
rsqAccelMax = max(rsqAccel)
rsqAccelAvg = mean(rsqAccel)
rsqAccelStd = std(rsqAccel)

% curve fitting against all time
figure
% midcross(abs(fvelttx.Vx), fs);
% hold on
scatter(t, abs(fvelttx.Vx), 'filled', 'DisplayName', 'Filtered Vx')
% hold on
% scatter(decelt, ypoly1, 'filled', 'DisplayName', 'Polyfit Vx')
hold on
scatter(accelt, ypoly2, 'filled', 'DisplayName', 'Polyfit Vx')
xlabel('Time [s]')
ylabel('V_T_o_t_a_l [mm/s]')
title('Calculated vs Approximated V_T_o_t_a_l over time')
xlim padded
ylim padded
grid on
% legend('Location', 'best')


%% simulated power ramp data


% simulated adjusted powers based on constant normalized enthalpy/PVratio
vcmd = 800; % mm/s
pcmd = 250; % W
psim1 = pcmd .* (sqrt(abs(ypoly2 / vcmd)));
% psim1 = pcmd .* (abs(ypoly2 / vcmd));

% check against constant power
pconst = pcmd * ones(length(t), 1);

% reduced y arrays for psim size
ysim1 = y(accelts, :);

% reduced x arrays for psim size
xsim1 = x(accelts, :);

% laser power 2D with laser state & simulated adjusted power superimposed
figure
scatter(x, y, [], p, 'filled', 'DisplayName', 'Experimental Power')
hold on
scatter(off(:, 2), off(:, 1), [], 'black', 'filled', 'DisplayName', 'Position-Laser
Off');
hold on
scatter(xsim1, ysim1, [], psim1, 'filled', 'DisplayName', 'Simulated Power');
xlabel('X-Position [mm]')
ylabel('Y-Position [mm]')
zlabel('Power [W]')
colormap(turbo)
hc = colorbar;
ylabel(hc,'W', 'FontSize', 16);
title('Laser power map')
```

```
xlim padded
ylim padded
legend('Location', 'best')

% laser power over time for comparing to ramped tests
figure
scatter(t, pconst, 'filled', 'DisplayName', 'Experimental Power');
hold on
scatter(accelt, psim1, 'filled', 'DisplayName', 'Simulated Power');
xlabel('Time [s]')
ylabel('Power [W]')
title('Laser power over time')
xlim padded
ylim padded
legend('Location', 'best')

% elapsed script time
toc
```

PCD Analysis Script

```
%% script setup


clear
close all
clc

% elapsed script time
tic

% import parameters
delimiterIn = (' ');
headerlinesIn = 27;
file1 = 'corrected Constant Power Guide Beam Test.pcd';
file2 = 'corrected Power Ramp Guide Beam Test.pcd';
file3 = 'corrected 23-07-21-p200v1000-0.05.pcd';
file4 = 'corrected 23-07-24-control-job6-0.05.pcd';
file = file1;
format long

% % file specification and parsing
a = importdata(file, delimiterIn, headerlinesIn);
% i = 0;
% r = 1e5;
% a.data((i * r) + 1:((i + 1) * r) + 1, :);


%% variable initialization


% filter out excess data
volt = a.data(:, 8);
volt = circshift(volt, 8); %#ok<NASGU>
s = a.data(:, ...
    9);
s = circshift(s, 8);
state_clip1 = find(s >= 1);
state_clip2 = find(s < 1);
state_start = state_clip1(1);
state_end = state_clip1(end);
a.data = a.data((state_start):(state_end - 8), :);

% inverted correction factors
% note pcd axes are swapped due to gloal and local co-ord differences
xoffset = 40.5;
yoffset = 82;
zoffset = -4;
xscaling = 1;
yscaling =  1;
zscaling = 1;

% data seperation
```

```
t = a.data(:, 1);
x = (a.data(:, 2)) / xscaling - xoffset;
y = (a.data(:, 3)) / yscaling + yoffset;

% create rotation matrix
theta = 90; % to rotate 90 counterclockwise
R = [cosd(theta) -sind(theta); sind(theta) cosd(theta)];
% rotate your point(s)
coords = [x y]'; % arbitrarily selected
rotcoords = R * coords;
x = (rotcoords(1, :))';
y = (rotcoords(2, :))';

z = (a.data(:, 4)) / zscaling + zoffset;
pyro1 = a.data(:, 5);
pyro2 = a.data(:, 6);
volt = a.data(:, 8);
c = 0.0189;
b = 0.0379;
p = ((volt / 0.4) - b) ./ (1000 * c);
s = a.data(:, 9);
state = [x, y, s];
zero = state(:, 3) == 0;
one = state(:, 3) == 1;
off = state;
off(one, :) = [];
on = state;
on(zero, :) = [];

% constants
timesteps = 0.00001;
fs = 1e5;
T = t * timesteps * 0.1;

% x velocity and acceleration
vx = diff(x) / timesteps;
Vx = [0; vx];
% ax = diff(Vx(:, 1)) / timesteps;
% Ax = [0; ax];

% y velocity and acceleration
vy = diff(y) / timesteps;
Vy = [0; vy];
% ay = diff(Vy(:, 1)) / timesteps;
% Ay = [0; ay];

% total velocity and acceleration
vtotal = sqrt((Vx.^2) + (Vy.^2));
veltt = array2timetable(vtotal, 'SampleRate', fs);
powtt = array2timetable(p, 'SampleRate', fs);

% timetable conversion
% veltty = array2timetable(Vy, 'SampleRate', fs);
velttx = array2timetable(Vx, 'SampleRate', fs);
% accelttx = array2timetable(Ax, 'SampleRate', fs);
```

```
% acceltty = array2timetable(Ay, 'SampleRate', fs);

p = circshift(p, 0);
fveltt = lowpass(veltt, 49, Steepness = 0.85);
fvelttx = lowpass(velttx, 61, Steepness = 0.85);
pvrx = p ./ abs(fvelttx.Vx);
pvrx(pvrx > 1) = 1;
% normalized enthalpy SS304
A = 0.525; % assumed from meso-nanosecond article for 316L
% A = 0.25; % a novel experimental method for in situ strain measurement during
selective laser melting
% P = 200; % commanded power watts
P = p; % actual power J/s
rho = 7.79e6; % estimated density g/m^3 K. C. Mills
C = 0.80; % estimated specific heat capacity J/gK K. C. Mills
Tm = 1454; % Tliq deg C K. C. Mills
D = 5.5e-6; %Tliq m^2/s K. C. Mills
% u = 1000 / 1e3; % commanded scan speed mm/s
u = abs(fveltt.vtotal) / 1e3; % total scan speed m/s
u = abs(fvelttx.Vx) / 1e3; % x-component scan speed m/s
r = 76e-6 / 2; % m from MIDI beam analysis average results table 1 D4sig
Hnorm = A * P ./ (pi * rho * C * Tm * sqrt(D * u * r^3));

% 5 vector times for power and velocity, p/v/t scatter p/v/e maps
v = abs(fvelttx.Vx);
% RP VSV2
rt = [1.76827 1.77279; 1.77396 1.77787; 1.77906 1.78302; 1.78417 1.78813; 1.78932
1.79328];
% RP VSV1
% rt = [1.48609 1.49058; 1.4918 1.49566; 1.49693 1.50077; 1.50205 1.5059; 1.5072
1.51105];

% CP times, convoluted, just use velocity bounds
deltaT = zeros(5,1);
deltaDeltaT = zeros(4,1);
for i = 1:5
    deltaT(i,:) = rt(i,2) - rt(i,1);
end
for i = 1:4
    deltaDeltaT(i,:) = rt(i+1,1) - rt(i,2);
end
% CP VSV2
% deltaDeltaT = [1.01017; deltaDeltaT];
% CP VSV1
deltaDeltaT = [1.07943; deltaDeltaT];

ct = zeros(5,2);
for i = 1:5
    ct(i,1) = ct(i,1) + deltaDeltaT(i,:);
    ct(i,2) = ct(i,1) + deltaT(i,:);
    if i > 1
        ct(i,1) = ct(i-1,2) + deltaDeltaT(i,:);
        ct(i,2) = ct(i,1) + deltaT(i,:);
    end
end
```

```
% VSV1
rt = rt.*fs - (1.76826*fs);
% ct = uint32(ct.*fs) - uint32(1.01016*fs);

% VSV2
% rt = rt.*fs - (1.48608*fs);
% cT = ct;
ct = uint32(ct.*fs) - uint32(1.07942*fs);

pAvg = [];
for i = 1:5
    avg = mean(p(rt(i,1):rt(i,2)));
    pAvg = [pAvg; avg];
end
pMean = mean(pAvg)

vAvg = [];
for i = 1:5
    avg = mean(v(rt(i,1):rt(i,2)));
    vAvg = [vAvg; avg];
end
vMean = mean(vAvg)

pError = -100 * ((200 - pMean) / 200)
vError = -100 * ((1000 - vMean) / 1000)


%% plotting


% posPlot(T, x)
% posPlot(T, y)
% posPlot(T, z)
% posTimePlot(x, y, T)
% laserStateMap(on, off)
% laserStatePlot(T, s)
laserPowerMap(x, y, p)
scannerVelocityMap(x, y, fvelttx.Vx)
% linearEnergyDensityMap(x, y, pvrx)
normalizedEnthalpyMap(x, y, Hnorm)
powerVelocityPlot(T, p, fvelttx.Vx)

% % unfiltered velocity power spectrum
% figure
% [pp,f] = pspectrum(velttx);
% scatter(f,pow2db(pp))
% xlabel('Frequency [Hz]')
% ylabel('Power Spectrum [dB]')
% title('Unfiltered Vx frequency domain')
% xlim padded
% ylim padded
% legend('Location', 'best')
%
% % unfiltered velocity spectrogram
```

```
% figure
% pspectrum(velttx, "spectrogram")

% elapsed script time
toc

%% function definitions


% calibrated & corrected x position over time
% function posPlot(T, x)
%     figure
%     scatter(T, x, 'filled', 'DisplayName', 'X-position')
%     xlabel('Time [s]')
%     ylabel('X-position [mm]')
%     title('X-position over time')
%     xlim padded
%     ylim padded
%     % legend('Location', 'best')
% end
%
% time-position 3D map
% function posTimePlot(x, y, T)
%     figure
%     scatter3(x, y, T, [], T, 'filled', 'DisplayName', 'Time');
%     xlabel('X-Position [mm]')
%     ylabel('Y-Position [mm]')
%     zlabel('Time [s]')
%     view(0, 90)
%     colormap(turbo)
%     hc = colorbar;
%     ylabel(hc,'Seconds', 'FontSize', 16);
%     title('Time-Position map;')
%     xlim padded
%     ylim padded
%     % legend('Location', 'best')
% end



% % laser state over time
% figure
% scatter(T, s, 'filled', 'DisplayName', 'Laser State');
% grid on
% xlabel('Time [s]')
% ylabel('State [on/off]')
% title('Laser state over time')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % laser power over time
% figure
% scatter(T, p, 'filled', 'DisplayName', 'Laser Power');
% xlabel('Time [s]')
```

```
% ylabel('Power [W]')
% title('Laser power over time')
% xlim padded
% ylim padded
% grid on
% legend('Location', 'best')

% laser state 2D map
% function laserStateMap(on, off)
%     figure
%     scatter(on(:, 1), on(:, 2), 'filled', 'DisplayName', 'Position-Laser On');
%     hold on
%     scatter(off(:, 1), off(:, 2), 'filled', 'DisplayName', 'Position-Laser Off');
%     grid on
%     xlabel('Y-Position [mm]')
%     ylabel('X-Position [mm]')
%     title('Laser state position map')
%     xlim padded
%     ylim padded
%     % legend('Location', 'best')
% end

% % laser state 2D map only on positions
% figure
% scatter(on(:, 1), on(:, 2), 'filled', 'DisplayName', 'Position-Laser On');
% xlabel('X-Position [mm]')
% ylabel('Y-Position [mm]')
% title('Laser state position map')
% % xlim([1 6.1])
% % ylim([11 12.2])
% % pbaspect([5 1 1])
% legend('Location', 'best')

% laser power map
% plowlim = 0;
% p(p <= plowlim) = nan;
function laserPowerMap(x, y, p)
    figure
    scatter3(x, y, p, [], p, 'filled', 'DisplayName', 'Power');
    set(gca,'FontSize', 14)
    xlabel('X-Position [mm]')
    ylabel('Y-Position [mm]')
    zlabel('Power [W]')
    view(0, 90)
    colormap(turbo)
    hc = colorbar;
    ylabel(hc,'Watts', 'FontSize', 14);
    title('Laser power map', 'FontSize', 20)
    xlim padded
    ylim padded
    % legend('Location', 'best')
end

% % unfiltered velocity over time
% figure
```

```
% scatter(T, vtotal, 'filled', 'DisplayName', 'Raw Vtotal')
% xlabel('Time [s]')
% ylabel('Velocity [mm/s]')
% title('Unfiltered total velocity over time')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % unfiltered X-velocity over time
% figure
% scatter(T, Vx, 'filled', 'DisplayName', 'Raw Vx')
% xlabel('Time [s]')
% ylabel('Velocity [mm/s]')
% title('Unfiltered Xvelocity over time')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % unfiltered velocity power spectrum
% figure
% [pp,f] = pspectrum(veltt);
% scatter(f,pow2db(pp))
% xlabel('Frequency [Hz]')
% ylabel('Power Spectrum [dB]')
% title('Unfiltered Vtotal frequency domain')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % unfiltered velocity power spectrum
% figure
% [pp,f] = pspectrum(velttx);
% scatter(f,pow2db(pp))
% xlabel('Frequency [Hz]')
% ylabel('Power Spectrum [dB]')
% title('Unfiltered Vx frequency domain')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % unfiltered velocity spectrogram
% figure
% pspectrum(veltt, "spectrogram")

% f0vx = lowpass(velttx, 61);
% % f0vy = lowpass(veltty, 61);
% fveltt = lowpass(veltt, 49, Steepness = 0.85);
% fvelttx = lowpass(velttx, 61, Steepness = 0.85);
% fveltt = bandpass(veltt, [35225 35593]);
% unfiltered velocity power spectrum
% figure
% [pp,f] = pspectrum(powtt);
% scatter(f,pow2db(pp))
% xlabel('Frequency [Hz]')
% ylabel('Power Spectrum [dB]')
```

```
% title('Unfiltered power frequency domain')
% xlim padded
% ylim padded
% legend('Location', 'best')

% fpowtt = lowpass(powtt, 49, Steepness = 0.95);
% % filtered velocity over time & power spectrum
% figure
% scatter(T, powtt.p, 'filled', 'DisplayName', 'Raw Power')
% hold on
% scatter(T, fpowtt.p, 'filled', 'DisplayName', 'Filtered Power')
% xlabel('Time [s]')
% ylabel('Power [W]')
% title('Raw vs Filtered Power over time')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % total velocity
% % speedlim = 9e3;
% vtotal = sqrt((f0vx.Vx.^2) + (f0vy.Vy.^2));
% % vtotal(vtotal >= speedlim) = nan;

% % filtered velocity over time & power spectrum
% figure
% subplot(2, 1, 1)
% scatter(T, veltt.vtotal, 'filled', 'DisplayName', 'Raw Vtotal')
% hold on
% scatter(T, fveltt.vtotal, 'filled', 'DisplayName', 'Filtered Vtotal')
% xlabel('Time [s]')
% ylabel('Velocity [mm/s]')
% title('Raw vs Filtered Vtotal over time')
% xlim padded
% ylim padded
% legend('Location', 'best')
% subplot(2, 1, 2)
% [p1,f1] = pspectrum(veltt);
% [p2,f2] = pspectrum(fveltt);
% scatter(f1, pow2db(p1), 'filled', 'DisplayName', 'Raw Vtotal')
% hold on
% scatter(f2, pow2db(p2), 'filled', 'DisplayName', 'Filtered Vtotal')
% xlabel('Frequency [Hz]')
% ylabel('Power Spectrum [dB]')
% title('Raw vs Filtered Vtotal power spectrum')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % filtered velocity over time
% figure
% scatter(T, fveltt.vtotal, 'filled', 'DisplayName', 'Filtered Vtotal')
% xlabel('Time [s]')
% ylabel('Velocity [mm/s]')
% title('Filtered total velocity over time')
% xlim padded
```

```
% ylim padded
% legend('Location', 'best')

% % filtered velocity over time
% figure
% scatter(T, fvelttx.Vx, 'filled', 'DisplayName', 'Filtered Vx')
% xlabel('Time [s]')
% ylabel('Velocity [mm/s]')
% title('Filtered x-velocity over time')
% xlim padded
% ylim padded
% legend('Location', 'best')

% laser velocity map
function scannerVelocityMap(x, y, V)
    figure
    scatter3(x, y, V, [], abs(V), 'filled', 'DisplayName', 'Velocity');
    set(gca,'FontSize', 14)
    xlabel('X-Position [mm]')
    ylabel('Y-Position [mm]')
    zlabel('Velocity [mm/s]')
    view(0, 90)
    colormap(turbo)
    hc = colorbar;
    ylabel(hc,'mm/s', 'FontSize', 14);
    title('Laser velocity map', 'FontSize', 20)
    xlim padded
    ylim padded
    % legend('Location', 'best')
end

% % laser velocity map
% figure
% scatter3(x, y, fveltt.vtotal, [], fveltt.vtotal, 'filled', 'DisplayName',
'Velocity');
% xlabel('X-Position [mm]')
% ylabel('Y-Position [mm]')
% zlabel('Velocity [mm/s]')
% view(0, 90)
% colormap(turbo)
% hc = colorbar;
% ylabel(hc,'mm/s', 'FontSize', 16);
% title('Laser velocity map')
% xlim padded
% ylim padded
% legend('Location', 'best')

% PVX ratio map
% p(:, :) = 275;
% function linearEnergyDensityMap(x, y, pvrx)
%     figure
%     scatter3(x, y, pvrx, [], pvrx, 'filled', 'DisplayName', 'PVX ratio');
%     xlabel('X-Position [mm]')
%     ylabel('Y-Position [mm]')
%     zlabel('PVX ratio [J/mm]')
```

```
%     view(0, 90)
%     colormap(turbo)
%     hc = colorbar;
%     clim([0, 1])
%     ylabel(hc,'J/mm', 'FontSize', 16);
%     title('PVX ratio map')
%     xlim padded
%     ylim padded
%     % legend('Location', 'best')
% end

% % PVTotal ratio map
% pvrtotal = p ./ abs(fveltt.vtotal);
% figure
% scatter3(x, y, pvrtotal, [], pvrtotal, 'filled', 'DisplayName', 'PVTotal ratio');
% xlabel('X-Position [mm]')
% ylabel('Y-Position [mm]')
% zlabel('PVTotal ratio [J/mm]')
% view(0, 90)
% colormap(turbo)
% hc = colorbar;
% clim([0, 1])
% ylabel(hc,'J/mm', 'FontSize', 16);
% title('PVTotal ratio map')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % filtered velocity over time & power spectrum
% figure
% subplot(2, 1, 1)
% scatter(T, velttx.Vx, 'filled', 'DisplayName', 'Raw Vx')
% hold on
% scatter(T, f0vx.Vx, 'filled', 'DisplayName', 'Filtered Vx')
% xlabel('Time [s]')
% ylabel('Velocity [mm/s]')
% title('Raw vs Filtered Vx over time')
% xlim padded
% ylim padded
% legend('Location', 'best')
% subplot(2, 1, 2)
% [p1,f1] = pspectrum(velttx);
% [p2,f2] = pspectrum(f0vx);
% scatter(f1, pow2db(p1), 'filled', 'DisplayName', 'Raw Vx')
% hold on
% scatter(f2, pow2db(p2), 'filled', 'DisplayName', 'Filtered Vx')
% xlabel('Frequency [Hz]')
% ylabel('Power Spectrum [dB]')
% title('Raw vs Filtered Vx power spectrum')
% xlim padded
% ylim padded
% legend('Location', 'best')

% % filtered velocity spectrogram
% figure
```

```
% pspectrum(fveltt, "spectrogram")

% % enthalpy over time
% figure
% scatter(T, Hnorm, 'filled', 'DisplayName', 'Enthalpy');
% xlabel('Time [s]')
% ylabel(' ')
% title('Normalized enthalpy over time')
% xlim padded
% ylim padded
% grid on

% % normalzied enthalpy emap
% hlowlim = 2;
% huplim = 4;
% clip1 = 1.48335 * fs;
% clip2 = 1.7218 * fs;
% clip1 = 1.5 * fs;
% clip2 = 1.7218 * fs;
% hnorm = Hnorm(clip1:clip2, :);
% hnorm(hnorm <= hlowlim) = nan;
% hnorm(hnorm >= huplim) = nan;
% xclip = x(clip1:clip2, :);
% yclip = y(clip1:clip2, :);

function normalizedEnthalpyMap(x, y, Hnorm)
figure
    % scatter3(xclip, yclip, hnorm, [], hnorm, 'filled', 'DisplayName', 'Enthalpy');
    scatter3(x, y, Hnorm, [], Hnorm, 'filled', 'DisplayName', 'Enthalpy');
    set(gca,'FontSize', 14)
    xlabel('X-Position [mm]')
    ylabel('Y-Position [mm]')
    zlabel('Normalized Enthalpy')
    view(0, 90)
    colormap(turbo)
    hc = colorbar;
    clim([0, 12]);
    ylabel(hc,'Normalized Enthalpy', 'FontSize', 14);
    title('Enthalpy map', 'FontSize', 20)
    xlim padded
    ylim padded
    % legend('Location', 'best')
end


% % laser state over time
% function laserStatePlot(T, s)
%     figure
%     scatter(T, s, 'filled', 'DisplayName', 'Laser State');
%     grid on
%     xlabel('Time [s]')
%     ylabel('State [on/off]')
%     title('Laser state over time')
%     xlim padded
%     ylim padded
```

```
%     % legend('Location', 'best')
% end

% % laser power over time
% figure
% scatter(T, p, 'filled', 'DisplayName', 'Laser Power');
% xlabel('Time [s]')
% ylabel('Power [W]')
% title('Laser power over time')
% xlim padded
% ylim padded
% grid on
% legend('Location', 'best')

% laser power and velocity over time
function powerVelocityPlot(T, p, V)
    figure
    format long
    yyaxis left
    scatter(T, p, 'filled', 'DisplayName', 'Laser Power');
    xlabel('Time [s]')
    ylabel('Power [W]')
    yyaxis right
    % scatter(T, vtotal, 'filled', 'DisplayName', 'Total Velocity');
    % scatter(T, fveltt.vtotal, 'filled', 'DisplayName', 'Total Velocity');
    scatter(T, abs(V), 'filled', 'DisplayName', 'X-Velocity');
    % p.DataTipTemplate.DataTipRows(1).Format = '%g'; % x
    % p.DataTipTemplate.DataTipRows(2).Format = '%g'; % y
    ylabel('Velocity [mm/s]')
    set(gca,'FontSize', 14)
    title('Laser power & velocity over time', 'FontSize', 20)
    xlim padded
    ylim padded
    grid on
    % for i = 1:5
    %     for j = 1:2
    %         xline(ct(i,j));
    %     end
    % end
    % legend('Location', 'best')
end

% % x position and velocity over time
% figure
% yyaxis left
% scatter(T, x, 'filled', 'DisplayName', 'X-Position');
% xlabel('Time [s]')
% ylabel('X-Position [mm]')
% yyaxis right
% % scatter(T, fveltt.vtotal, 'filled', 'DisplayName', 'Total Velocity');
% scatter(T, fvelttx.Vx, 'filled', 'DisplayName', 'X-Velocity');
% ylabel('Velocity [mm/s]')
% title('X-position & velocity over time')
% xlim padded
% ylim padded
```

```
% grid on
% legend('Location', 'best')

% % laser power over time
% figure
% scatter(T, p, 'filled', 'DisplayName', 'Laser Power');
% xlabel('Time [s]')
% ylabel('Power [W]')
% title('Laser power over time')
% xlim padded
% ylim padded
% grid on
% legend('Location', 'best')
%
% % laser velocity over time
% figure
% % scatter(T, fveltt.vtotal, 'filled', 'DisplayName', 'Total Velocity',
'MarkerFaceColor','#D95319');
% scatter(T, fvelttx.Vx, 'filled', 'DisplayName', 'X-Velocity',
'MarkerFaceColor','#D95319');
% xlabel('Time [s]')
% ylabel('Velocity [mm/s]')
% title('Laser velocity over time')
% xlim padded
% ylim padded
% grid on
% legend('Location', 'best')

% T = T(1:50000, :);
% p = p(1:50000, :);
% vtotal = vtotal(1:50000, :);
% s = s(1:50000, :);
% Hnorm = Hnorm(1:50000, :);
% figure
% plot(T, vtotal, 'linewidth', 2)
% addaxis(T, p, 'linewidth', 2)
% addaxis(T, s, 'linewidth', 2)
% addaxis(T, Hnorm, 'linewidth', 2)
% grid on
% legend('Location', 'best')
% legend({'Velocity', 'Power', 'State', 'Normalized Enthalpy'})

% % velocity map
% figure
% scatter3(x, y, f0vx.Vx, [], f0vx.Vx, 'filled', 'DisplayName', 'Velocity');
% xlabel('X-Position [mm]')
% ylabel('Y-Position [mm]')
% zlabel('Velocity [mm/s]')
% view(0, 90)
% colormap(turbo)
% hc = colorbar;
% ylabel(hc,'mm/s', 'FontSize', 16);
% title('Velocity map')
% xlim padded
% ylim padded
```

```
% legend('Location', 'best')

% f0ax = bandpass(accelttx, [49 1697]);
% f0ay = bandpass(acceltty, [49 1697]);

% % total acceleration
% % accellim = 9e3;
% atotal = sqrt((f0ax.Ax.^2) + (f0ay.Ay.^2));
% % vtotal(vtotal >= speedlim) = nan;

% % laser acceleration over time
% figure
% scatter(T, atotal, 'filled', 'DisplayName', 'Total Acceleration',
'MarkerFaceColor','#D95319');
% xlabel('Time [s]')
% ylabel('Acceleration [mm/s^2]')
% title('Laser accelreation over time')
% xlim padded
% ylim padded
% grid on
% legend('Location', 'best')
```

Build File Script

```
"""
This script is part of the End of Vector Defect Mitigation project.

The CLIModifier script enables generation of CLI files with CLI+
functionality for adjusting inter-vector power on Aconity LPBF systems.

The geometry section is configured for a single type. Running with
adjusted jump/mark delays and SW off results in depression mitigation.
Running with SW on and 0us extension time enables continuous marking.

Author:  Tony York
Contact: ajyork@miner.utep.edu
"""
# imported modules ############################################################


# used to track execution time
from timeit import default_timer as timer
start = timer()

# used to determine rounding precision and rounding float values
import math as m

# used to convert lists and perform operations on arrays
import numpy as np

# used for scatter plots
from matplotlib import pyplot as plt

# used for color maps
import matplotlib.cm as cm


# machine parameters 'global constants' ###############################################


# lower power cutoff limit [W] 'int', laser 1
LOW_POWER_LIM = 50
# upper power cutoff limit [W] 'int', laser 1
HIGH_POWER_LIM = 500
# length of timesteps [s] 'float'
# based on xy2-100 protocol transmission frequency
```

```
CONTROLLER_CYCLE = 1e-5
# scanner speed limits [mm/s] 'int'
# https://www.raylase.de/en/products/prefocusing-deflection-units/axialscan-fi
# ber.html
# depending on tuning scanner dynamics are different: H = 9.5, VC = 15.9 [m/s]
SCAN_SPEED_LIM = 9500
# bed dimensions for vector size/offsets/initial position
# position limits [mm] 'float'
POSITION_LIM = 125.0
# length of timestep [us] 'int'
TIMESTEP_LENGTH = int(1e6 * CONTROLLER_CYCLE)
# for single power type this write line is constant
POWER_LINE = 7
# for single geometry type this write line is constant
POLY_LINE = 8


# function prototypes ##########################################################


def calculate_vectors(numofvectors):
    """
    Calculates the vector global variables for computing powers & coordinates.

    Parameters:

    ramptimes: array [int]
        Ramp length affects the ramp gradient.
    powerdeltas: array [int]
        Difference in power affects the ramp gradient.

    Returns:

    numofvectors: int
        The number of vectors to be scanned.
    vectors: array [int]
        Array of numbered vectors equal in amount to numofvectors.
    vectorsperpower: int
        How many vectors will be run for one pair of powers.
    vectorsperramptime: int
        How many vectors will be run for one ramp length.
    """
    vectors = np.arange(0, numofvectors)
    return vectors
```

```
def calculate_rounding_precision(initial_value):
    # Calculate the number of decimal places required for rounding
    precision = max(0, -int(m.floor(m.log10(initial_value))))
    return precision


# used for calculating various microvector parameters
def calculate_microvectors(cmdscanspeed, xdist, numofvectors):
    """
    Calculates the microvector variables for computing powers & coordinates.

    Parameters:

    cmdscanspeed: int
        Commanded scan speed.
    xdist: int
        Vector length along x-axis.
    numofvectors: int
        The number of vectors to be scanned.

    Returns:

    mvlength: float
        Microvector length is the most precise unit for this program.
    powerprecision: int
        Precision used for power values.
    positionprecision: int
        Precision used for coordinates.
    microvectors: int
        Number of decompositions of a vector based on controller cycle.
    totalvectors: int
        Number of microvectors in a layer.
    """
    mvlength = cmdscanspeed * CONTROLLER_CYCLE
    powerprecision = 0
    positionprecision = calculate_rounding_precision(mvlength)
    microvectors = int(round((xdist + mvlength) / mvlength))
    totalvectors = numofvectors * microvectors
    return mvlength, powerprecision, \
           positionprecision, microvectors, totalvectors


# used for generating x coordinates
def calculate_xcoords(x0, xdist, vectors, microvectors, positionprecision,
xoffset, mvlength):
```

```
    """
    Computes x-coordinates.

    Parameters:

    x0: float
        Starting position for vector along x-axis.
    xdist: float
        Vector length along x-axis.
    vectors: array [int]
        Array of numbered vectors equal in amount to numofvectors.
    positionprecision: int
        Precision used for coordinates.

    Returns:

    xcoords: NDArray [float]
        Array of coordinates along the x-axis composing the scanpath.
    """
    xcoords = []
    for i in vectors:
        if i % 2 == 0:
            start = x0 + (i * xoffset)
            end = (xdist + x0) + (i * xoffset)
            length = np.linspace(start, end, microvectors)
            length = length.tolist()
            xcoords.extend(length)
        else:
            start = (xdist + x0) + (i * xoffset)
            end = x0 + (i * xoffset)
            length = np.linspace(start, end, microvectors)
            length = length.tolist()
            xcoords.extend(length)
    xcoords = np.array(xcoords)
    xcoords = np.round(xcoords, decimals = positionprecision)
    return xcoords


# used for generating y coordinates
def calculate_ycoords(y0, ydist, vectors, microvectors, positionprecision):
    """
    Computes y-coordinates.

    Parameters:
```

```
    y0: float
        Starting position for vector along y-axis.
    ydist: float
        Vector length along y-axis, hatch spacing.
    vectors: array [int]
        Array of numbered vectors equal in amount to numofvectors.
    microvectors: int
        Number of decompositions of a vector based on controller cycle.
    positionprecision: int
        Precision used for coordinates.

    Returns:

    ycoords: NDArray [float]
        Array of coordinates along the y-axis composing the scanpath.
    """
    ycoords = []
    for i in vectors:
        start = y0 + (i  * ydist)
        for i in range(microvectors):
            ycoords.append(start)
    ycoords = np.array(ycoords)
    ycoords = np.round(ycoords, decimals = positionprecision)
    return ycoords


# used for generating powers
def calculate_powers(a1, b1, c1, d1, a2, b2, c2, d2, a3, b3, c3, d3, e3, mu1,
mu2, mu3, mu4, mu5, mu6, t1, t2, t3, \
                     cmdpower, cmdscanspeed, powerprecision, vectors, \
                     numofvectors, rampoffset1, rampoffset2, microvectors,
numofpowers1, numofpowers2):
    """
    Computes powers.

    Parameters:

    vectorsperramptime: int
        How many vectors will be run for one ramp length.
    vectorsperpower: float
        How many vectors will be run for one pair of powers.
    ramppowers: array [int]
        Array of commanded power values.
    ramptimes: array [int]
        Ramp length affects the ramp gradient.
```

```
    powerprecision: int
        Precision used for power values.
    microvectors: int
        Number of decompositions of a vector based on controller cycle.
    numofvectors: int
        The number of vectors to be scanned.
    rampoffset: int
        Number of timesteps [microvectors] ramp is offset.


    Returns:

    powers: NDArray [int]
        Array of power values for each microvector.
    """
    # Vx approx
    # T1 = (t1 - mu1) / mu2
    # T2 = (t2 - mu3) / mu4
    T3 = (t3 - mu5) / mu6

    # for _ in t1:
    #     t1 = a1 * pow(T1, 3)
    #     t2 = b1 * pow(T1, 2)
    #     t3 = c1 * T1
    # ypoly1 = np.array([[t1], [t2], [t3]])
    # ypoly1 = ypoly1.sum(axis = 0)
    # ypoly1 = ypoly1 + d1

    # print('ypoly1 = ', ypoly1)

    # for _ in t2:
    #     t4 = a2 * pow(T2, 3)
    #     t5 = b2 * pow(T2, 2)
    #     t6 = c2 * T2
    # ypoly2 = np.array([[t4], [t5], [t6]])
    # ypoly2 = ypoly2.sum(axis = 0)
    # ypoly2 = ypoly2 + d2

    for _ in t3:
        t7 = a3 * pow(T3, 4)
        t8 = b3 * pow(T3, 3)
        t9 = c3 * pow(T3, 2)
        t10 = d3 * T3
    ypoly3 = np.array([[t7], [t8], [t9], [t10]])
    ypoly3 = ypoly3.sum(axis = 0)
```

```
    ypoly3 = ypoly3 + e3

    # pactual1 = cmdpower * (pow((abs(0.8 * ypoly1 / cmdscanspeed)), 0.5))
    # # pactual1 = cmdpower * (pow((abs(0.8 * ypoly1 / cmdscanspeed)), 1))
    # # pactual1 = cmdpower * (pow((abs(1.0 * ypoly1 / cmdscanspeed)), 1))
    # power1 = pactual1
    # # print('power 1 = ', power1)

    # pactual2 = cmdpower * (pow((abs(0.8 * ypoly2 / cmdscanspeed)), 0.5))
    # # pactual2 = cmdpower * (pow((abs(0.8 * ypoly2 / cmdscanspeed)), 1))
    # # pactual2 = cmdpower * (pow((abs(1.0 * ypoly2 / cmdscanspeed)), 1))
    # power2 = pactual2

    pactual3 = cmdpower * (pow((abs(1.0 * ypoly3 / cmdscanspeed)), 0.5))
    # # pactual3 = cmdpower * (pow((abs(0.8 * ypoly3 / cmdscanspeed)), 1))
    # power3 = pactual3

    # power1 = pactual1[0]
    # power1[power1 < LOW_POWER_LIM] = LOW_POWER_LIM
    # power1[power1 > cmdpower] = cmdpower
    # power1 = np.round(power1, decimals = powerprecision)
    # power1 = power1.astype(int)
    # # print('power 1 = ', power1)

    # power2 = pactual2[0]
    # power2[power2 < LOW_POWER_LIM] = LOW_POWER_LIM
    # power2[power2 > cmdpower] = cmdpower
    # power2 = np.round(power2, decimals = powerprecision)
    # power2 = power2.astype(int)

    power3 = pactual3[0]
    power3[power3 < LOW_POWER_LIM] = LOW_POWER_LIM
    power3[power3 > cmdpower] = cmdpower
    power3 = np.round(power3, decimals = powerprecision)
    power3 = power3.astype(int)

    # powers = np.array([])
    # for i in vectors:
    #     for i in power2:
    #         powers = np.append(powers, i)
    #     for i in range(microvectors - numofpowers1 - numofpowers2):
    #         powers = np.append(powers, cmdpower)
    #     for i in power1:
    #         powers = np.append(powers, i)
```

```
    powers = np.array([])
    for i in vectors:
        for i in range(microvectors - numofpowers1):
            powers = np.append(powers, cmdpower)
        for i in power3:
            powers = np.append(powers, i)


    # powers = powers.astype(int)
    # powerspervector = int(len(powers) / (numofvectors * 2))
    # offsetpowers = powers
    # powers = np.array([])
    # for i in range(numofvectors * 2):
    #     powersoffset = offsetpowers[powerspervector * i:powerspervector \
    #                    * (i + 1):]
    #     if i % 2 == 0:
    #         powersoffset = np.roll(powersoffset, rampoffset1)
    #     else:
    #         powersoffset = np.roll(powersoffset, rampoffset2)
    #     powers = np.append(powers, powersoffset)
    powers = powers.astype(int)
    powers = np.roll(powers, rampoffset1)
    return powers


def plot_power_ramp(powers):
    time = range(len(powers))
    plt.figure(1)
    plt.scatter(time, powers)
    plt.grid()
    plt.xlabel('Timesteps [10us]', fontsize = 15)
    plt.ylabel('Power [W]', fontsize = 15)
    plt.title('Power Ramp', fontsize = 20)


def plot_scatter_powers(xcoords, ycoords, powers):
    """
    Plots power values for each xy coordinate pair.

    Parameters:

    ycoords: NDArray [float]
        Array of coordinates along the y-axis composing the scanpath.
    ycoords: NDArray [float]
        Array of coordinates along the y-axis composing the scanpath.
```

```
    powers: NDArray [int]
        Array of power values for each microvector.

    Returns:

    None
    """
    plt.figure(2)
    plt.scatter(xcoords, ycoords, c = powers, cmap = 'turbo')
    plt.xlabel('Xcmd [mm]', fontsize=15)
    plt.ylabel('Ycmd [mm]', fontsize=15)
    plt.title('XY Power Map', fontsize=20)
    cbar = plt.colorbar()
    cbar.set_label('Watts', fontsize=15)
    plt.grid(True)


# formatting coordinates based on CLI
def merged_coords(xcoords, ycoords, numofvectors):
    """
    Merges xy coordinates to satisfy file format.

    Parameters:

    ycoords: array [float]
        Array of coordinates along the y-axis composing the scanpath.
    ycoords: array [float]
        Array of coordinates along the y-axis composing the scanpath.

    Returns:

    mergedcoords: str
        String of coordinates joined by commas.

    """
    xcoords = np.split(xcoords, numofvectors)
    ycoords = np.split(ycoords, numofvectors)
    [l.tolist() for l in xcoords]
    [l.tolist() for l in ycoords]
    mergedcoords = []
    for i in range(len(xcoords)):
        coords = [val for pair in zip(xcoords[i], ycoords[i]) for val in \
                        pair] + xcoords[len(ycoords):] + \
                        ycoords[len(xcoords):]
        mergedcoords.append(coords)
```

```
    return mergedcoords


def merged_powers(powers, numofvectors):
    """
    Merges power values to satisfy file format.

    Parameters:

    powers: array [int]
        Array of power values for each microvector.

    Returns:

    mergedpowers: str
        String of power values/indices joined by commas.
    """
    powersindex = np.arange(0, len(powers))
    powers = np.split(powers, numofvectors)
    powersindex = np.split(powersindex, numofvectors)
    [l.tolist() for l in powers]
    [l.tolist() for l in powersindex]
    mergedpowerz = []
    for i in range(len(powersindex)):
        mergedpowers = [val for pair in zip(powersindex[i], powers[i]) for val in
\
                        pair] + powersindex[len(powers):] + \
                        powers[len(powersindex):]
        mergedpowerz.append(mergedpowers)
    return mergedpowers, mergedpowerz


def writing_header(path):
    """
    Writes header info in the CLI file.

    Parameters:

    path: str
        Directory where the CLI file will be written.

    Returns:

    None
```

```
    """
    # cli header/footer format information for single layer scans
    header = ['$$HEADERSTART', '$$ASCII', '$$UNITS/1', '$$VERSION/200', \
              '$$HEADEREND', '$$GEOMETRYSTART', '$$LAYER/0.05000']
    # writing header info
    with open(path, 'w') as f:
        for line in header:
            f.write(line)
            f.write('\n')


def writing_scan_strat(path, mergedpowers, totalvectors, mergedcoords,
numofvectors):
    with open(path, 'a') as file:
        for i in range(numofvectors):
            lines = []
            linez = []
            for _, sublist in enumerate(mergedpowers):
                line = ','.join(map(str, sublist))
                lines.append(line)
                continue
            file.write('$$POWERS/' + lines[0] + '\n')
            for _, sublist in enumerate(mergedcoords):
                line = ','.join(map(str, sublist))
                linez.append(line)
                continue
            # file.write(f'$$POLYLINE/1,2,{microvectors},' + linez[i] + '\n')
            file.write(f'$$POLYLINE/1,2,{int(totalvectors / numofvectors)},' +
linez[i] + '\n')


def writing_footer(path):
    """
    Writes footer info in the CLI file.

    Parameters:

    path: str
        Directory where the CLI file will be written.

    Returns:

    None

    """
```

```
    footer = ['$$GEOMETRYEND']
    with open(path, 'a') as f:
        f.write(footer[0])

def build_visualizer(path):
    """
    Plots power values for each xy coordinate pair.

    Parameters:

    ycoords: NDArray [float]
        Array of coordinates along the y-axis composing the scanpath.
    ycoords: NDArray [float]
        Array of coordinates along the y-axis composing the scanpath.
    powers: NDArray [int]
        Array of power values for each microvector.

    Returns:

    None
    """
    header = '$$HEADERSTART'
    footer = '$$GEOMETRYEND'
    newlayer = '$$LAYER/'
    newhatch = '$$HATCHES/'
    newpower = '$$POWERS/'
    newpoly = '$$POLYLINE/'
    footer = '$$GEOMETRYEND'
    sep1 = '/'
    sep2 = ','

    layernum = []
    layerline = []
    layer = []

    powernum = []
    powerline = []
    powers = []

    polynum = []
    polyline = []
    x = []
    y = []

    with open(path, 'r') as f:
```

```
        lines = f.readlines()
        if header in lines[0] and footer in lines[-1]:
            for index, line in enumerate(lines):
                if footer in line:
                    end = index
                    print('\n' + f"header ends at line {index}" + '\n')
                    continue
                if newlayer in line:
                    layernum.append(index)
                    layerline.append(line)
                    print(f"\n\nnew layer at line {index}" + '\n\n')
                elif newpower in line:
                    powernum.append(index)
                    line = line.replace("\n", "")
                    powerline.append(line)
                    print(f"new powers at line {index}" + '\n')
                elif newpoly in line:
                    polynum.append(index)
                    line = line.replace("\n", "")
                    polyline.append(line)
                    print(f"new poly at line {index}" + '\n')
        else:
            print(f'something\'s fishy with your file :3' + '\n\n')


    for i in range(len(powerline)):
        power1 = powerline[i].split(sep1)
        power1 = power1[1].split(sep2)
        powerindex = power1[::2]
        power = power1[1::2]
        powers.append(power)
    powers = [item for sublist in powers for item in sublist]
    powers = list(map(int, powers))


    for i in range(len(polyline)):
        poly1 = polyline[i].split(sep1)
        poly1 = poly1[1].split(sep2)
        partid3 = poly1[0]
        dir1 = poly1[1]
        numofvectors3 = poly1[2]
        poly = poly1[3:len(poly1)]
        p1x = poly[::2]
        x.append(p1x)
        p1y = poly[1::2]
```

```
        y.append(p1y)
    x = [item for sublist in x for item in sublist]
    x = list(map(float, x))
    y = [item for sublist in y for item in sublist]
    y = list(map(float, y))

    plt.figure(4)
    plt.scatter(x, y, c = powers, cmap = 'turbo')
    plt.xlabel('Xcmd [mm]', fontsize=15)
    plt.ylabel('Ycmd [mm]', fontsize=15)
    plt.title('XY Power Map', fontsize=20)
    cbar = plt.colorbar()
    cbar.set_label('Watts', fontsize=15)
    plt.grid(True)


def main():

    # build parameters 'local variables' ########################################


    # commanded speed [mm/s] 'int'
    cmdscanspeed = 1000
    if 0 < cmdscanspeed > SCAN_SPEED_LIM:
        print('\n', 'invalid value for cmdscanspeed, must be between 0 & 9500',
            '\n')

    # commanded power [W] 'int'
    cmdpower = 200
    if cmdpower > HIGH_POWER_LIM:
        print('\n', 'invalid value for cmdpower, cannot exceed highpowerlim',
            '\n')

    # initial x position [mm], 'float'
    x0 = 0 # -8.5 # -2.5
    if x0 > POSITION_LIM:
        print('\n', 'invalid value for x0, cannot exceed poslim', '\n')

    # vector length along x-axis [mm], 'float'
    xdist = 5
    if xdist > POSITION_LIM:
        print('\n', 'invalid value for xdist, cannot exceed poslim', '\n')

    # initial y position [mm], 'float'
    y0 = 0 # 9.54 # 0.54
```

```
    if y0 > POSITION_LIM:
        print('\n', 'invalid value for y0, cannot exceed poslim', '\n')

    # vector length along y-axis [mm], 'float'
    ydist = -1.2e-1
    if ydist > POSITION_LIM:
        print('\n', 'invalid value for ydist, cannot exceed poslim', '\n')

    # shift along x-axis [mm], 'float'
    xoffset = 2.0e-2 # 1.25e-2
    if xoffset > POSITION_LIM:
        print('\n', 'invalid value for xoffset, cannot exceed poslim', '\n')

    # length of end ramp [timesteps], 'int'
    numofpowers1 = 118 # 51 # 95 # 108
    if numofpowers1 >= ((xdist / cmdscanspeed) / CONTROLLER_CYCLE):
        print('\n', 'invalid value for numofpowers1, cannot exceed \
            +/-((xdist / cmdscanspeed) / CONTROLLER_CYCLE)', '\n')

     # length of start ramp [timesteps], 'int'
    numofpowers2 = 0 # 93 #51 # 93
    if numofpowers2 >= ((xdist / cmdscanspeed) / CONTROLLER_CYCLE):
        print('\n', 'invalid value for numofpowers2, cannot exceed \
            +/-((xdist / cmdscanspeed) / CONTROLLER_CYCLE)', '\n')

    # shift of ramp start [timesteps], 'uint'
    rampoffset1 = 69 # 65
    if rampoffset1 >= ((xdist / cmdscanspeed) / CONTROLLER_CYCLE):
        print('\n', 'invalid value for numofpowers, cannot exceed \
            +/-((xdist / cmdscanspeed) / CONTROLLER_CYCLE)', '\n')

    # shift of ramp start [timesteps], 'uint'
    rampoffset2 = 0
    if rampoffset2 >= ((xdist / cmdscanspeed) / CONTROLLER_CYCLE):
        print('\n', 'invalid value for numofpowers, cannot exceed \
            +/-((xdist / cmdscanspeed) / CONTROLLER_CYCLE)', '\n')

    # number of mark vectors [vectors], 'int'
    numofvectors = 52
    if numofvectors < 1:
        print('\n', 'invalid value for numofvectors, must be 1 or greater', '\n')


    # script parameters 'global variables' ##################################
```

```
    vectors = calculate_vectors(numofvectors)

    mvlength, powerprecision, positionprecision, \
    microvectors, totalvectors \
        = calculate_microvectors(cmdscanspeed, xdist, numofvectors)

    # generating xcoordinates
#######################################################


    xcoords = calculate_xcoords(x0, xdist, vectors, microvectors,
positionprecision, xoffset, mvlength)


    # generating ycoordinates
#######################################################


    ycoords = calculate_ycoords(y0, ydist, vectors, microvectors,
positionprecision)


    # generating powers
#############################################################


    a1 = 36.5124
    b1 = -18.5275
    c1 = -375.7918
    d1 = 589.0672
    a2 = -37.5067
    b2 = 8.9787
    c2 = 376.9938
    d2 = 451.4505
    mu1 = 48
    mu2 = 27.5681
    mu3 = 47
    mu4 = 26.9907
    t1 = np.arange(1,96)
    t2 = np.arange(1, 94)
    # t1 = np.arange(32, 83)
    # t2 = np.arange(24, 75)

    # Vx/Vtotal
    # a3 = -70.6455
    # b3 = -2.1014
```

```
    # c3 = 352.3616
    # d3 = -11.6979
    # e3 = 380.7499
    # mu5 = 54.5
    # mu6 = 31.3209
    # t3 = np.arange(1, 109)

    # Vtotal/Vtotal
    # a3 = -26.7436
    # b3 = 40.7588
    # c3 = 187.3973
    # d3 = -156.7569
    # e3 = 468.8716
    # mu5 = 46.5
    # mu6 = 26.7021
    # t3 = np.arange(1, 93)

    # original
    a3 = -50.7891
    b3 = 104.7891
    c3 = 246.4193
    d3 = -889.2183
    e3 = -314.3189
    mu5 = 63.5
    mu6 = 36.5171
    t3 = np.arange(1, 119)

    powers = calculate_powers(a1, b1, c1, d1, a2, b2, c2, d2, a3, b3, c3, d3, e3,
mu1, mu2, mu3, mu4, mu5, mu6, t1, \
                              t2, t3, cmdpower, cmdscanspeed, powerprecision,
vectors, \
                              numofvectors, rampoffset1, rampoffset2, \
                              microvectors, numofpowers1, numofpowers2)


    # verifying scan strategy ##################################################


    plot_power_ramp(powers)
    # plot_scatter_powers(xcoords, ycoords, powers)
    plt.show()

    # formatting build data ####################################################
```

```
    mergedcoords = merged_coords(xcoords, ycoords, numofvectors)

    mergedpowers, mergedpowerz = merged_powers(powers, numofvectors)


    # file setup ##################################################################


    path = (r'C:\Users\Tony\Documents\pythonscripts\clirepository\sectioning\mo'
        r'delsection_001_s1_vs.cli')
    writing_header(path)
    writing_scan_strat(path, mergedpowerz, totalvectors, mergedcoords,
numofvectors)
    writing_footer(path)


    # checking file ###############################################################


    # build_visualizer(path)
    # plt.show()

if __name__ == "__main__":
    main()

end = timer()
print('program execution time = ' + str(end - start) + 's')
```

VITA

Antonio [Tony] J. York was born in El Paso, Texas. He moved to several cities in Texas before returning to study at El Paso Community College and eventually the University of Texas at El Paso where he completed his B.S.M.E as a first-generation student in May 2021. Since 2021 he has been conducting research at the W.M. Keck Center focusing on control of laser powder bed fusion systems. He had the opportunity to pursue an internship and co-op during the summer and fall of 2022 with an AM startup, Tailored Alloys, in Youngstown, Ohio where he had the pleasure of researching hardware and software available for metal AM printers. Upon completion of his M.S.M.E program he seeks to transition into UTEP's CS program to pursue multidisciplinary specialization in fields related to his research.

Contact Information: Antonio James York - ajyork@miners.utep.edu