THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENT OF MECHANICAL AND NUCLEAR ENGINEERING

POROSITY DETECTION AND ANALYSIS USING MATLAB OF THIN-WALLED STRUCTURES FABRICATED USING POWDER BED FUSION AT DIFFERENT ANGLES WITH RESPECT TO THE RECOATER BLADE

MATTHEW EDWARD DOLACK
SPRING 2017

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Mechanical Engineering
with honors in Mechanical Engineering

Reviewed and approved* by the following:

Timothy W. Simpson
Professor of Mechanical & Industrial Engineering
Thesis Supervisor

Sean Brennan
Professor of Mechanical & Nuclear Engineering
Honors Advisor

Edward W. Reutzel
Adjunct Professor of Engineering Science and Mechanics
Faculty Reader

* Signatures are on file in the Schreyer Honors College.

ABSTRACT

The additive manufacturing industry has undergone significant innovation and advancement in the field of selective laser melting technology during the past decade. A common concern when fabricating parts using powder bed fusion, which is a form of selective laser melting, is unintentional internal porosity. Unintentional porosity in a fabricated part can be problematic because it can negatively impact both its mechanical and heat transfer properties. When fabricating components using powder bed fusion there are many alterable process parameters that can affect internal porosity. A Ti-6Al-4V pad with 25 rectangular fins ranging in thickness from 300 to 60 microns was printed using powder bed fusion to test whether build angle with respect to the recoater blade affects internal porosity. The laser scan pattern for each unique fin thickness was not changed when printed at angles of 0°, 30°, 60°, and 90° with respect to the recoater blade. Data on laser spot size and corresponding melt pool size were also analyzed for possible effects on internal porosity. Using computerized tomography scanning, each fin pad's internal structure was digitized. The internal structure of each built fin was then analyzed for internal defects using a custom-built MATLAB program. Analysis of the collected defect data indicates that internal porosity of thin-walled structures can be decreased substantially if fabricated using a laser scan pattern and spacing that causes melt pool overlap to occur. Internal porosity of thin-walled structures is independent of fabrication angle with respect to the recoater blade when melt pool overlap occurs during part construction and is dependent on fabrication angle with respect to the recoater blade when melt pool overlap does not occur during part construction. Furthermore, when melt pool overlap does not occur during part fabrication

internal porosity is most significant in parts printed at 0° with respect to the recoater blade and is least significant in parts printed at 90° with respect to the recoater blade.

TABLE OF CONTENTS

ACRONYMS AND ABBREVIATIONS

PBF	Powder Bed Fusion
AM	Additive Manufacturing
CT	Computerized Tomography
CAD	Computer-Aided Design
ROI	Region of Interest

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to Dr. Timothy W. Simpson and Dr. Edward W. Reutzel for their support and direction during the completion of my thesis. Additionally, I would like to thank Gabi Gundermann for her insight, especially regarding laser scan patterns, during data analysis. The analyzed fin pad was designed by Corey Dickman and all CT scans were performed by Griffin Jones at Penn State's CIMP-3D. Lastly, without sponsorship and funding from America Makes partner Honeywell International Inc. much of the conducted research would not have been possible. Thank you.

Chapter 1

Introduction

Additive manufacturing (AM) is a term that describes the building of a three-dimensional object through a layer-by-layer deposition of a material. Some of the materials currently being researched and fabricated with are metals, polymers, composites and ceramics [Li, 2016]. AM is not a new technology, but it has experienced a period of significant growth during the past decade [Li, 2016]. One of the more recent innovations in AM is powder bed fusion (PBF), which is a form of selective laser melting. A description of PBF follows after which post-processing techniques, non-destructive computed tomography evaluation, and part porosity are discussed.

1.1: Powder Bed Fusion Additive Manufacturing

PBF AM builds a three-dimensional object layer-by-layer by selectively melting metal powder to form a solid mass. The powder contained in a build layer is melted using a high intensity laser beam or electron beam. The laser beam spot size is large in comparison to the relative size of the metal powder particles yet small in comparison to the area of a build layer. Figure 1 displays the layering and selective melting of metal powder during PBF.

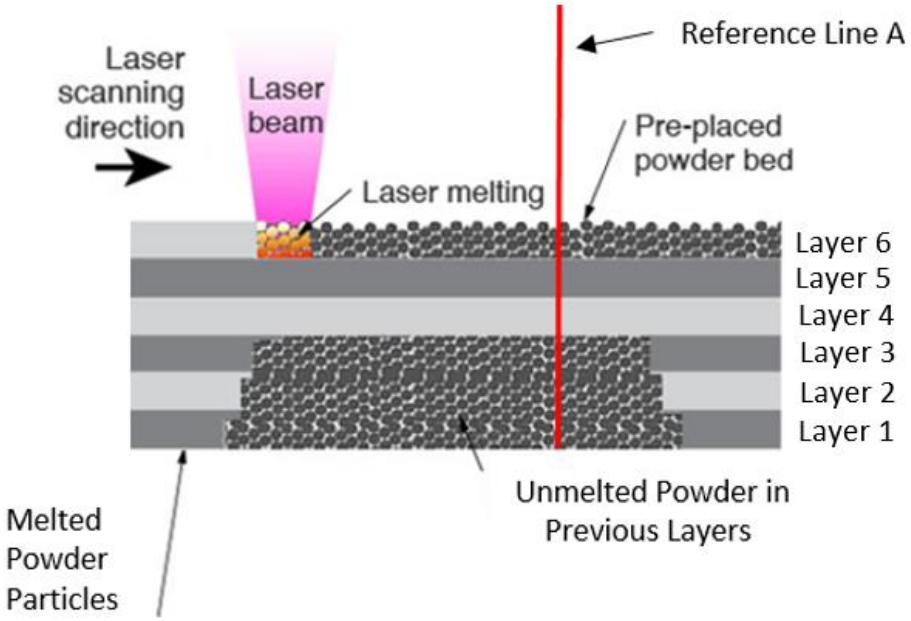


Figure 1: Layering and selective melting of metal powder during PBF [King, 2015].

Not all of the metal powder in a build layer is melted by the laser beam. Contained in Figure 1 is a reference line labeled ‘A’. Build Layers 1 through 6 intersect reference line ‘A’ at a single point. Along reference line ‘A’ the metal powder in Layer 4 and 5 has been solidified by a pass from the laser beam. Therefore, Layers 4 and 5 are solidified at line ‘A’ both as individual points within their respective layers, but also together because they share a common boundary. Layers 3 and 6 also share a common boundary with solidified Layers 4 and 5. However, because Layers 3 and 6 were not melted by the laser beam, they are not physically connected to Layers 4 and 5 along reference line ‘A’. The laser beam that melts the metal powder is highly adjustable, with alterable parameters such as speed, intensity, and velocity.

A micro-view of a PBF layer is shown in Figure 2 and details the interaction between the metal powder and laser beam during printing. More specifically, Figure 2 displays a single powder layer that has received two passes, labeled Pass #1 and Pass #2, from the laser. When

metal powder comes into contact with the laser beam, it is first melted, forming what is known as a melt pool, and then cooled through convection and conduction to form a solidified mass. If the melt pools of two laser passes overlap each other, then they will become a conjoined, solidified mass. The region of overlap between the melt pool of Pass #1 and Pass #2 in Figure 2 is highlighted. The trough of the semi-elliptical melt pools from Pass #1 and Pass #2 extend into the previous build layer. PBF AM is able to produce three-dimensional objects layer-by-layer because the trough of the melt pool in the topmost layer extends into the preceding layer. Extension of the melt pool trough into the preceding layer allows build layers to solidify with one another in an additive manner. Whether or not the melt pool of adjacent laser passes overlap one another is dependent on hatch spacing and laser intensity. Hatch spacing is the distance that separates laser passes measured from the center of the laser beam, as seen in Figure 2. Laser scan pattern is the path the laser travels when selectively melting regions of a powder layer.

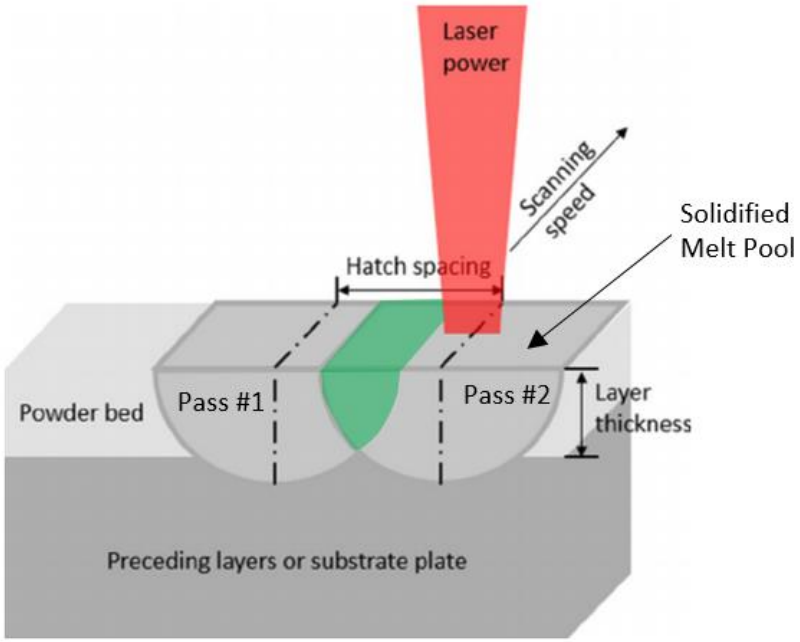


Figure 2: Micro-view of two overlapping PBF laser passes [Yap, 2015].

A macro-view of a PBF process is illustrated in Figure 3. Figure 1 and 2 depicted micro-views of PBF objects being fabricated. In Figure 3 the object being fabricated is labeled and located with an arrow. As per the iterative manufacturing method that is PBF, a new layer of metal powder must be spread over the build area at the start of each build layer. Each powder layer is spread over the build area using a recoater blade, which is shown in Figure 3.

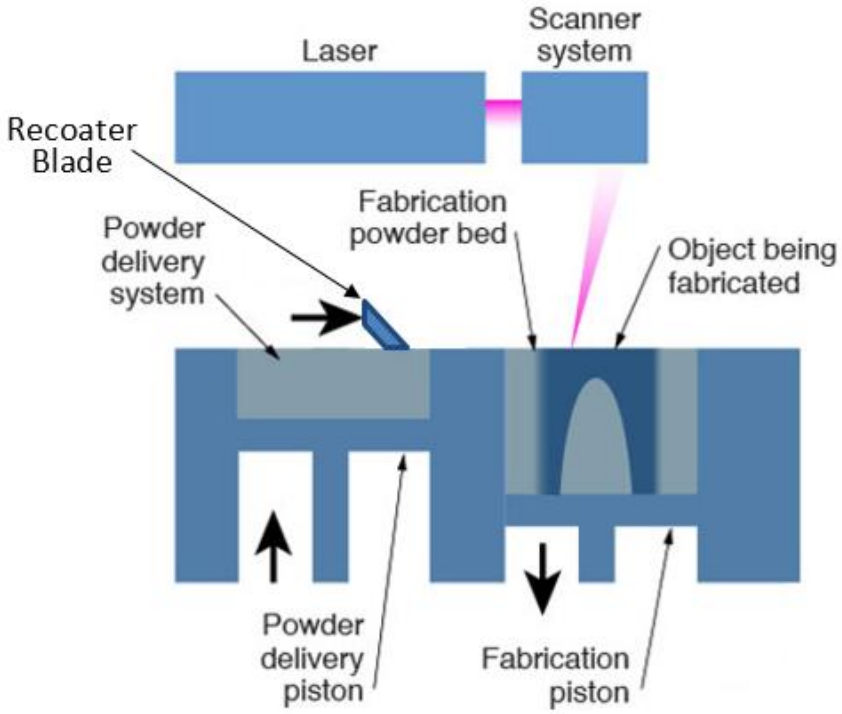


Figure 3: Macro-view of PBF process [King, 2015].

An important geometric characteristic regarding the recoater blade and object being fabricated is the angle relative to each other. Depending on how a part being fabricated is positioned on the build plate, the angle with respect to the recoater blade will vary.

After the completion of a PBF build the fabricated parts are typically stress relieved to minimize any residual internal stresses and then heat treated to ensure the parts are of the highest quality possible. The post-processing techniques of stress relieving and heat treating are discussed in the next section.

1.2: Post-Processing of PBF Components

After a PBF print job is complete, the fabricated parts can be removed from the build plate. Before the parts are removed, however, they should undergo a relieving process to reduce any residual stresses contained within them. Once stress relieved the fabricated parts can be removed from the build plate and their likelihood of either plastically yielding or prematurely failing while in service is considerably less [Mercelis, 2006; Withers, 2007]. With the parts removed from the build plate they can be heat treated using hot isostatic pressing to reduce the size of any internal defects, which will in turn increase density [Atkinson, 2000; Lu, 2015]. The post-processing technique of stress relieving is further described in the next section.

1.2.1: Residual Stress Relief in PBF Components

Residual stresses are stresses that remain in a fabricated part after it cools to room temperature [Mercelis, 2006]. During PBF printing, internal stresses form within the part being constructed because thermal gradients induced by the high intensity laser beam cause regions of the part to either thermally expand or contract [Mercelis, 2006]. Physical damage to both the build plate and constructed part can result from large residual stresses. In Figure 4, residual

stresses have caused a fabricated part to fracture and detach from the build plate. The reason for stress relieving PBF components is to minimize the magnitude of residual stresses.

Thermal stress relieving of parts constructed using PBF is done while the parts are still attached to the build plate. The printed parts and build plate are heated to a temperature well-below the melting point of their respective materials for a period of time typically spanning a few hours. Then, the fabricated parts and build plate are slowly cooled to room temperature. By following this process residual stresses are minimized. The purpose of stress relieving is not to affect the mechanical properties of the printed parts. Hot isostatic pressing, which is discussed in the next section, is used to affect the mechanical properties of the PBF printed parts.

Figure 4. Residual stresses within a part fabricated using PBF can be strong enough to cause it to fracture and detach from the build plate [Simpson, 2015].

1.2.2: Hot Isostatic Pressing of PBF Components

After minimizing the residual stresses within the PBF constructed parts through stress relief post-processing, the parts are ready to be removed from the build plate and undergo heat

treatment. The primary reason for heat treating metal fabricated parts is to optimize their mechanical properties. The type of heat treatment most commonly used to refine the material properties of PBF AM parts is hot isostatic pressing. When a part is subjected to hot isostatic pressing it is exposed to "a high pressure and elevated temperature in a specially constructed vessel" [Atkinson, 2000]. Exposing the fabricated parts to high pressure helps collapse any internal pores [Atkinson, 2000]. The elevated temperature causes both the collapsed pores as well as any unmelted metal powder within the constructed part to fuse [Atkinson, 2000]. Post-process heat treating of PBF fabricated parts through hot isostatic pressing is beneficial because it allows parts to approach near-full densification, which results in mechanical property and performance optimization.

After a PBF part has undergone post-processing, the internal structure of the fabricated part can be evaluated to determine if it will perform as engineered. Computerized tomography (CT) scanning is one method discussed in the next section that is commonly used to evaluate the internal structure of additively manufactured parts.

1.3: Computerized Tomography Evaluation of Additively Manufactured Components

When an AM build is completed, the printed parts can either enter service or undergo testing and evaluation to determine their mechanical properties and internal structure. The internal structure of a fabricated part can be evaluated either destructively, meaning the part is destroyed, or non-destructively, meaning the part is not destroyed. Computerized tomography (CT) scanning generates a digital three-dimensional rendering of an object by compiling thousands of two-dimensional X-rays captured at multiple angles [Noel, 2008]. A commercially

available industrial CT scanner made by General Electric is shown in Figure 5. Figure 6 contains a computer-aided design (CAD) rendering of a pad with thin rectangular fins on top. Shown in Figure 7 is an image of the same pad after fabrication using PBF AM.

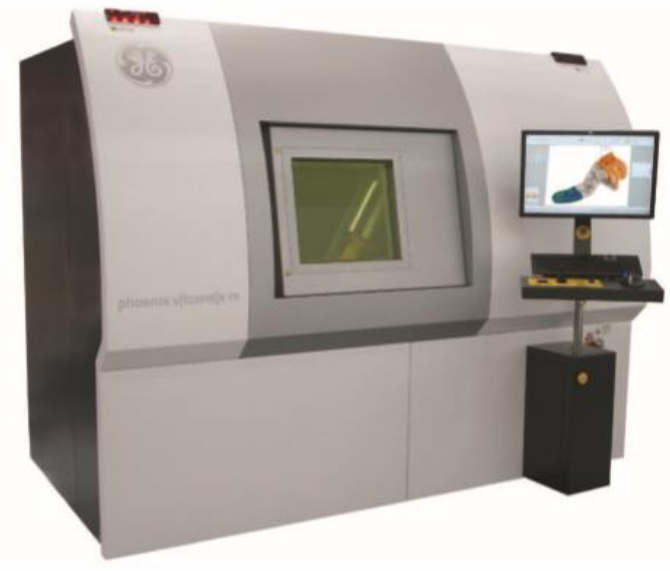


Figure 5: Industrial CT scanner made by General Electric Corporation [General Electric Company, 2016].

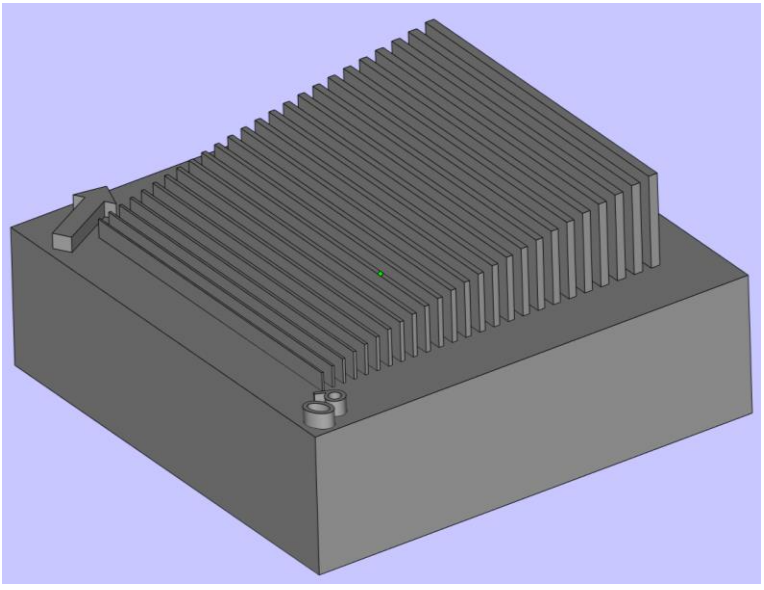


Figure 6: Computer-aided design (CAD) rendering of a pad with thin rectangular fins.

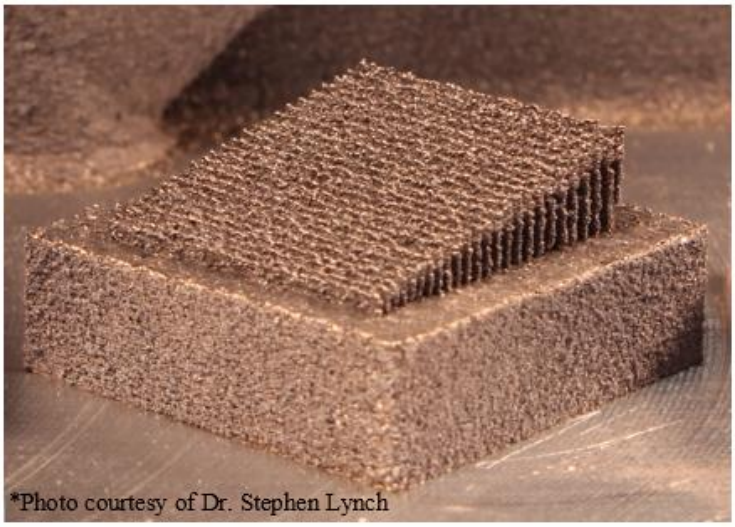


Figure 7: A pad with thin rectangular fins after fabrication using PBF.

CT scanning is a valuable form of non-destructive evaluation because both the external and internal structure of a scanned part are digitized. Figure 8 contains a CT rendering of a fin pad fabricated using PBF similar to the one shown in Figure 7. Additionally, the internal structure of one fin from the fin pad fabricated using PBF shown in Figure 7 is displayed in Figure 9. CT renderings are generated using voxels, which are similar to pixels except they are three-dimensional [Noel, 2008]. Therefore, the CT rendering shown in Figure 8 was generated using voxels while the internal fin structure displayed in Figure 9 was generated using pixels. Many defects, or instances of porosity, are contained within the fin body shown in Figure 9. The fin structure shown in Figure 10 is identical to the one displayed in Figure 9, except that all internal defects have been highlighted. It is important to understand the impact porosity can have on the structural integrity of a part, especially if the part is subjected to mechanical stresses. Porosity and what it will mean in the context of this research are discussed in next.

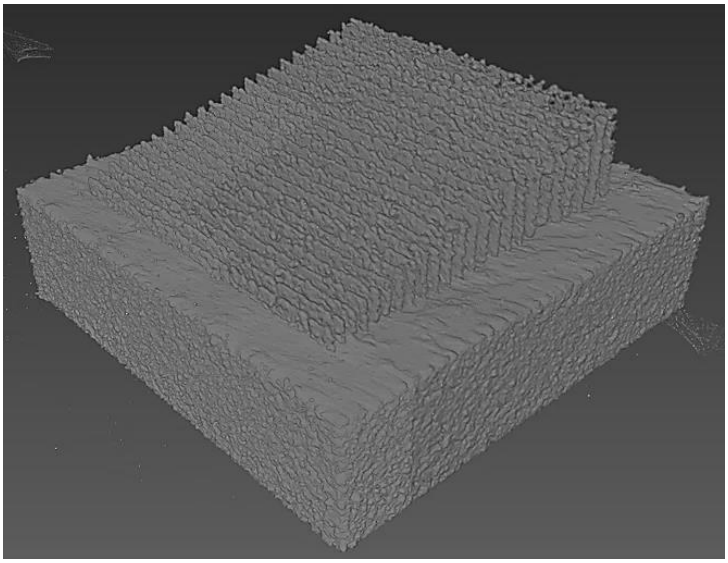


Figure 8: CT rendering of a PBF fabricated fin pad.

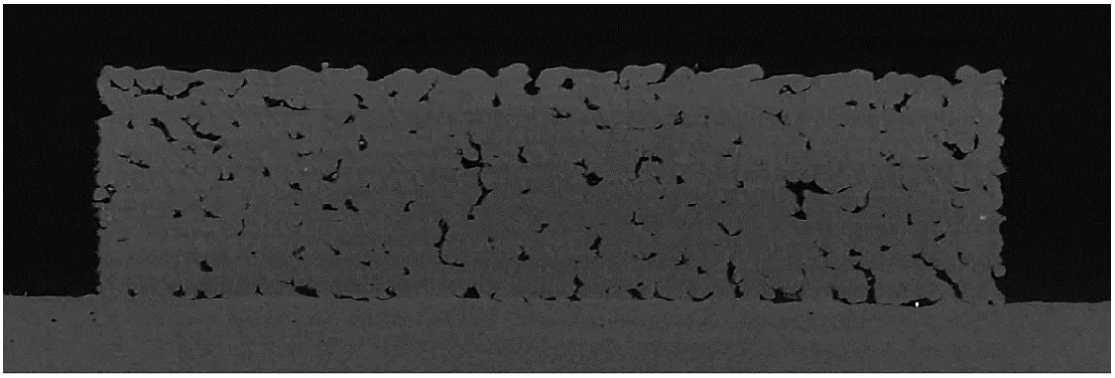


Figure 9: Internal structure of a fin.

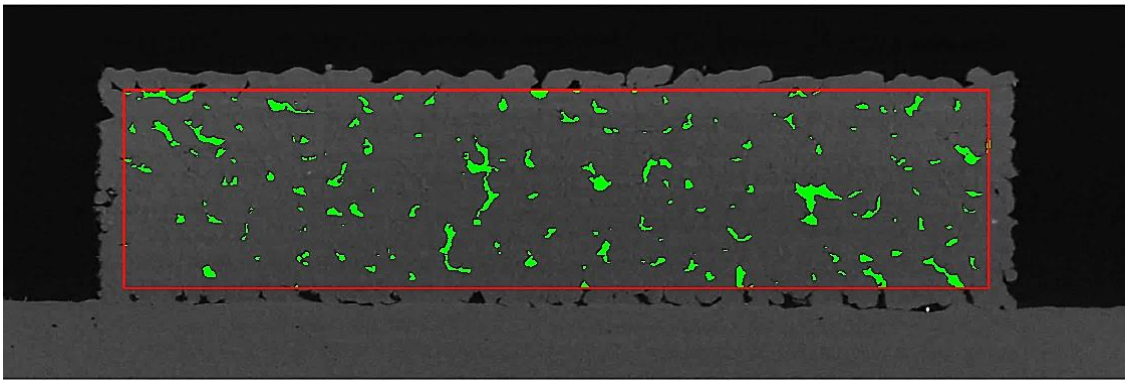


Figure 10: Internal structure of a fin with instances of porosity highlighted.

1.4: Porosity in PBF Fabricated Components

Internal porosity is a broad term used to describe the presence of pockets or voids inside of a fabricated part. The term porosity will be used in this document to refer to any unintentional pocket or void in a location within a part that should be uniformly solid. Figure 11 is a zoomed-in view of the internal fin structure shown in Figure 9. The dashed circles in Figure 11 identify instances of internal porosity. Limiting porosity in PBF fabricated parts is important because porosity can degrade a part's mechanical performance [Tang, 2017]. When the mechanical properties of a part are degraded, premature failure can result. Therefore, it is crucial that parts built using PBF are printed as per the CAD file specifications to ensure they perform as engineered. Published literature on the relationship PBF process parameters have on internal porosity is reviewed in the next chapter. The research direction of this thesis is also discussed.

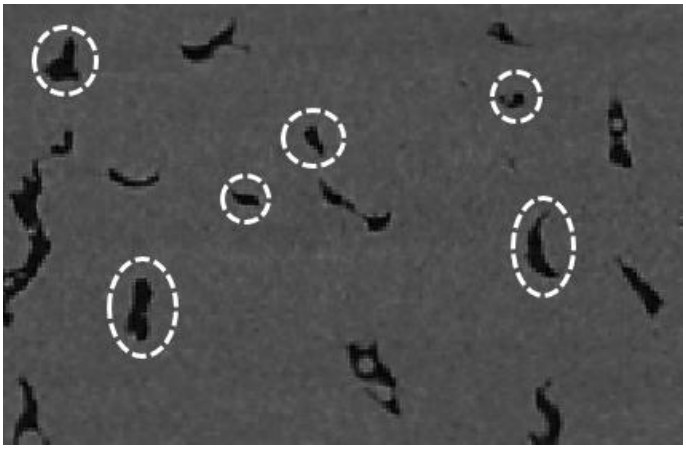


Figure 11: Zoomed-in view of internal fin structure with instances of porosity circled.

Chapter 2

Literature Review and Thesis Direction

Porosity in parts fabricated using PBF is an AM topic of great importance. Presently, extensive research into the relationship PBF process parameters have on internal defects is underway in both domestic and international academic institutions. Numerous PBF AM publications indicate that printing low-porosity, high-quality parts is achievable through optimization of the following process parameters: laser power, scanning speed, and powder layer thickness [Yadroitsev, 2015]. Other publications document the effect hatch spacing and melt pool overlap have on build porosity [Gong, 2014; Yap, 2015]. Methods for evaluating the effects that PBF parameters have on build porosity vary widely because of the fundamental differences many parameters have with one another. Therefore, definitively linking a set of parameters to the consistent fabrication of low-porosity, high-quality parts poses a significant challenge. An understudied process parameter whose effect on built porosity is not well understood is fabrication angle with respect to the recoater blade. A diagram illustrating part fabrication angle with respect to the recoater blade is displayed in Figure 12. Figure 12A symbolically represents part fabrication angle with respect to the recoater blade using the variable theta, Φ. Furthermore, Figure 12B and 12C illustrate fabrication angles with respect to the recoater blade of 90° and 0°, respectively.

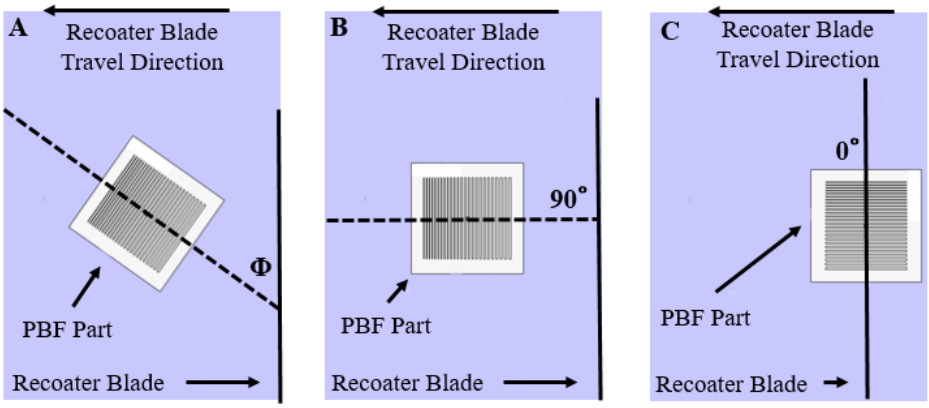


Figure 12: Illustration 'A' symbolically represents part fabrication angle, Φ, with respect to the recoater blade. Illustration 'B' and 'C' display what fabrication angles of 90° and 0°, respectively, look like with respect to the recoater blade for a part under construction.

To experimentally evaluate whether PBF part orientation with respect to the recoater blade has an effect on internal porosity a specialized test was developed. The developed test uses a custom-built pad containing 25 thin rectangular fins. The height of each fin decreases with decreasing cross-sectional thickness at a height-to-thickness ratio of 10. The pad was designed with fins of thin cross-sections because internal porosity is a particularly problematic development in small, design critical structures.

To test part orientation with respect to the recoater blade, one fin pad was built at each of the following angles: 0°, 30°, 60°, and 90°. All four fin pads were fabricated during the same print job using Ti-6Al-4V powder on an EOS Corporation EOSINT M280 laser PBF printer. The internal structure of all four fabricated fin pads were then non-destructively evaluated using CT scanning. The voxel rendering of each fin pad was then recharacterized as thousands of .TIFF images. Using the fin pad image stacks, a defect detection and evaluation program was created using the computing language MATLAB. The porosity data returned by the developed defect detect program for each fin pad was then plotted and analyzed. In addition to analyzing the porosity data for a potential relationship to recoater blade angle, both laser beam and melt pool

size were considered as well. In the next chapter the design considerations that went into creating and fabricating the fin pad are discussed.

Chapter 3

Fin Pad Development and Fabrication

To test whether recoater blade angle has an effect on internal porosity of thin-walled structures, a pad with 25 fins of decreasing cross-sectional thickness was designed. A side-view CAD rendering of the designed fin pad is shown in Figure 13. The fin pad was designed by Corey Dickman from the Pennsylvania State University Applied Research Lab. Each fin on the fin pad is assigned a number. Fin #1 corresponds to the tallest and thickest fin, and Fin #25 corresponds to the shortest and thinnest fin. The fin pad shown in Figure 13 has Fin #1 and Fin #25 labeled. Table 1 documents the thickness and height corresponding to each of the 25 fins on the pad. The height-to-thickness ratio for each fin is also outlined in Table 1.

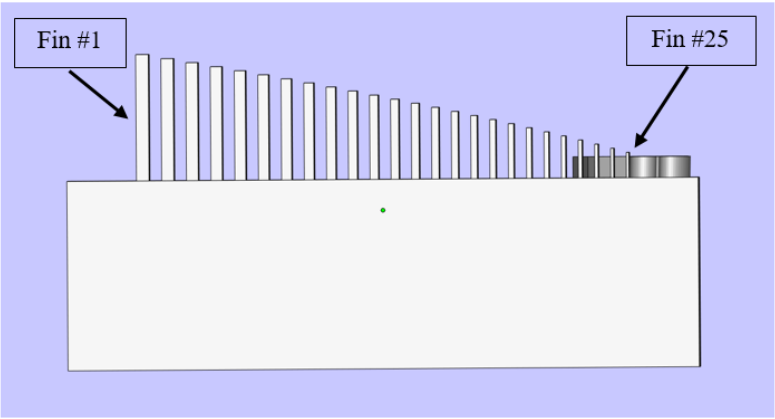


Figure 13: Side-view CAD rendering of the designed fin pad.

Table 1: Designed height, thickness and height-to-thickness ratio for each fin.

Fin No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Fin Height (mm)	3.0	2.9	2.8	2.7	2.6	2.5	2.4	2.3	2.2	2.1	2.0	1.9	1.8	1.7	1.6	1.5	1.4	1.3	1.2	1.1	1.0	0.9	0.8	0.7	0.6
Fin Thickness (mm)	0.3	0.29	0.28	0.27	0.26	0.25	0.24	0.23	0.22	0.21	0.20	0.19	0.18	0.17	0.16	0.15	0.14	0.13	0.12	0.11	0.10	0.09	0.08	0.07	0.06
Height to Thickness Ratio	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10

Each fabricated fin pad was printed at the unique angle of either 0°, 30°, 60°, or 90° with respect to the recoater blade. Figure 14 shows the travel direction and angle of the recoater blade in reference to each fin pad built. The laser scan pattern used to fabricate each fin pad was the same across all tested recoater blade angles and was determined using proprietary EOS Corporation software. Figure 15 shows the laser scan pattern used to fabricate each fin. Fin #1 through Fin #24 share the same laser scan pattern and hatch spacing, but with adjusted contour spacing to compensate for the decreasing cross-sectional thickness of each fin. The green lines in Figure 15 represent the path traveled by the laser during PBF fabrication and the blue lines represent the outline of the fin under construction. ΔX in Figure 15 identifies the width component of the inner contour rectangle used for the fabrication of Fin #1 through Fin #24. The width of the inner contour rectangle for each fin is documented in Table 2. Because of the extremely thin cross-section of Fin #25 its laser scan pattern is significantly different from all the other fins. Due to EOSINT M280 PBF printer limitations, Fins #22, #23, #24 and #25 failed to print properly on all four fabricated fin pads.

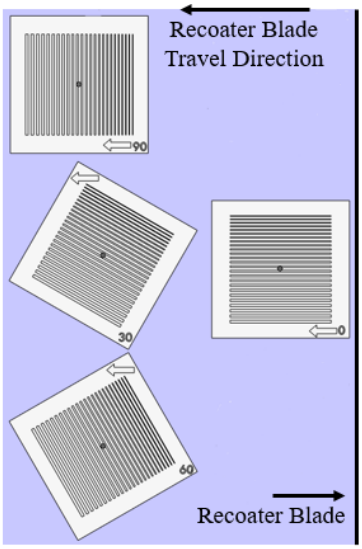


Figure 14: Overhead view of all four fabricated fin pads. Listed in the corner of each fin pad is its fabrication angle with respect to the recoater blade.

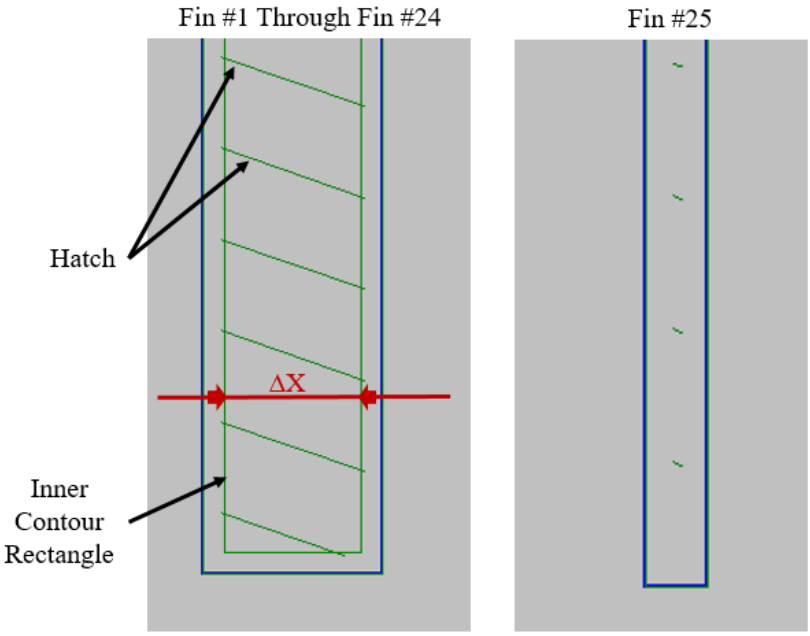


Figure 15. Laser scan pattern used for the fabrication of each fin. The width component of the inner contour rectangle is identified by Δx.

Table 2: Width component, Δx, of the inner contour rectangle used for fin construction.

Fin No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Width of Inner Hatching Rectangle (mm)	.244	.234	.220	.208	.198	.190	.183	.167	.159	.154	.142	.136	.125	.114	.102	.092	.082	.076	.059	.049	.045	.033	.022	.011	NA

After fabricating the four fin pads using PBF, each pad's internal structure was non-destructively evaluated using CT scanning. The CT renderings of each pad were then recharacterized as image stacks for use by the porosity detection and analysis program. Appendix A contains hyperlinks to access the image stacks for each fin pad. In the next chapter the design and development of the MATLAB porosity detection and analysis program is discussed.

Chapter 4

MATLAB Defect Detection Program Development

In this chapter, the design considerations that went into developing and implementing the MATLAB defect detection program are discussed. After digitizing the internal structure of each fin pad into thousands of .TIFF images, an automated method of analysis capable of systematically evaluating and quantifying the porosity of each fin was needed. Using the computing language MATLAB, a custom-built program was developed to return porosity data on a fin-by-fin basis. Required program capabilities are outlined in Table 3. The first required program capability listed in Table 3 is the ability to read and analyze image stacks that can contain thousands of images.

Table 3: Required program capabilities.

1. Analyze large image stacks.
2. Return an interactive side-view representation of the CT scanned fin pad.
3. Accept user input regarding which fins to analyze.
4. Process multiple fins at the same time.
5. Select image stills corresponding to the approximate fin center.
6. For all selected image stills, the MATLAB program will: 6.1 - Define a region of interest (ROI) within the fin. 6.2 - Calculate the area of the ROI. 6.3 - Highlight regions of porosity within the ROI. 6.4 - Calculate the percent porosity within the ROI. 6.5 - Calculate the total area void of material within the ROI. 6.6 - Determine the number of defects within the ROI. 6.7 - Classify voids within the ROI by size.
7. Export defect data to a Microsoft Excel spreadsheet.

To address the program capability requirement of analyzing large image stacks in a seamless manner, image cropping was used. Image cropping was used to force exploitable elements of uniformity into every image in the image stack. The element of uniformity forced into the image stack through cropping is sharp greyscale pixel transitions along edges coinciding with fin edges. The presence of sharp greyscale pixel transitions along fin edges means searching algorithms can be used to automatically detect the edges, or outer boundary, of a fin without requiring user input.

Because of the newly created sharp pixel transitions in areas corresponding to fin boundaries, all of the fin images in an image stack can be quickly analyzed using an unchanging set of edge finding algorithms. The ability to use an unchanging set of edge finding algorithms for every fin image satisfies Capability #1 from Table 3 of seamlessly analyzing large image

stacks. Figure 16 contains two image stills of fins that have been cropped differently. Figure 16A is properly cropped because all regions with sharp greyscale pixel transitions, i.e., a black to grey transition, correspond to a fin edge. The dashed outline in Figure 16A highlights the area where dark greyscale pixels are required for the MATLAB program to operate properly. Conversely, Figure 16B is not cropped properly because the two dashed rectangles border a sharp greyscale pixel transition area that corresponds to a build pad edge, not a fin edge. After properly cropping all four image stacks the program capabilities of returning an interactive side-view image of the CT scanned fin pad and accepting user input regarding which fins to analyze were addressed.

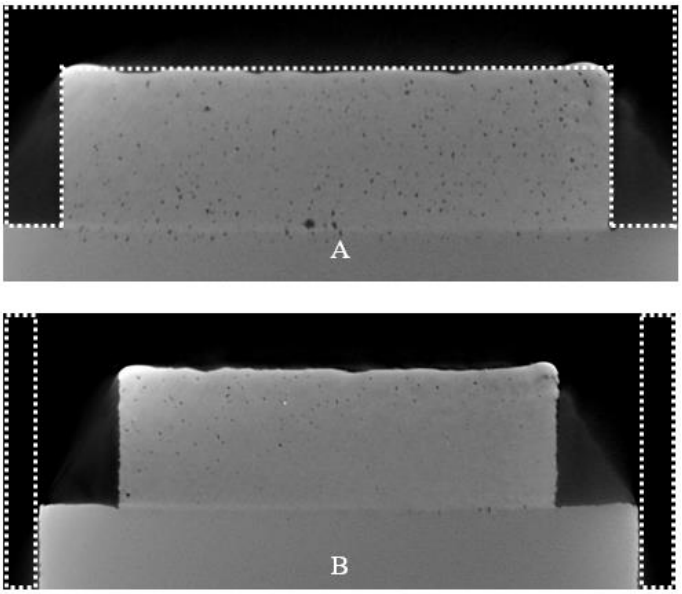


Figure 16: Fin image stills illustrating the areas that need to be included (A) and/or removed (B) via cropping for the developed MATLAB program to operate properly.

Capability #2 of an interactive side-view rendering and Capability #3 of user-controlled fin selection were required design constraints for the MATLAB program because they help streamline and automate the process of determining which fins to analyze. Using MATLAB

array summation functions and thresholding the number of pixels with a greyscale value above 60 is determined for every image in an image stack. Greyscale pixel values range from 0 to 255. A greyscale pixel with a value of 0 is considered black, and a pixel of value 255 is considered white. A greyscale thresholding value of 60 was selected and used because it separates regions containing material from regions void of material in a fin image still. A side-view rendering unique to the fin pad being analyzed is then generated by plotting the number of pixels with a greyscale value above 60 for each image still. Array summation and thresholding works because image stills without a fin present provide a baseline pixel count and the decreasing height of each fin pad fin results in a decreasing pixel count. For comparison, a CT rendering of a fin pad and its corresponding MATLAB side-view rendering can be seen in Figure 17. The MATLAB returned side-view rendering contains clearly identifiable fins that correspond closely to the CT rendering of the fabricated fin pad. Additionally, locations where fins are misshapen or malformed are easily discernable on the side-view rendering.

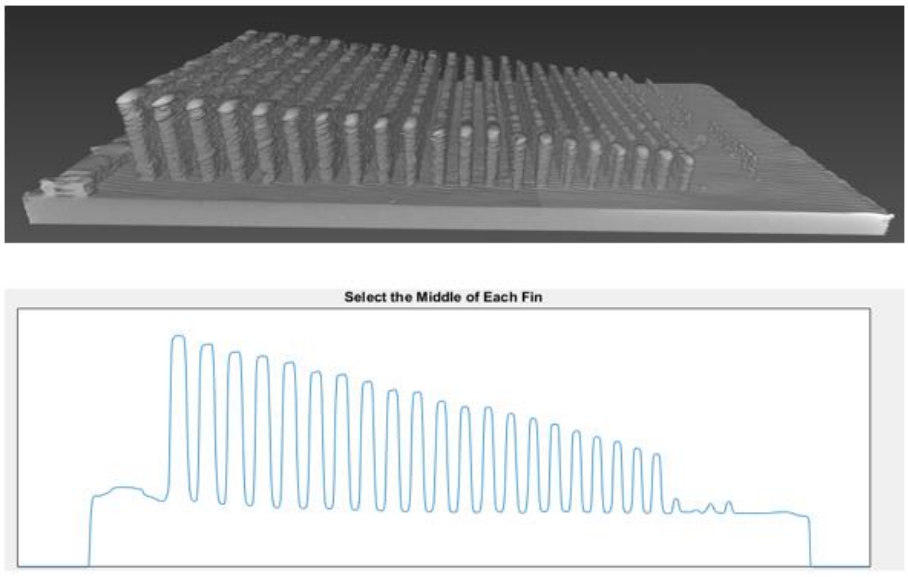


Figure 17: Side-view comparison of a CT scanned pad and a MATLAB returned rendering.

Using the MATLAB returned side-view rendering, the user is tasked with selecting which fins to analyze by clicking on each of their respective longitudinal centers. The vertical lines in Figure 18 show the approximate longitudinal centers of 21 fins selected for analysis. Moreover, once the user selects which fins on the fin pad to analyze, another side-view rendering is displayed with each fin numbered in order of selection. The numbering of each selected fin is also shown in Figure 18. The ability to select more than one fin for analysis satisfies program Capability #4 listed in Table 3. Additionally, Capability #5 is also satisfied, again from Table 3, because the user-selected longitudinal center of each fin is logged by the MATLAB program, and its corresponding image still is selected and retained for later analysis. After selecting all of the fins desired for porosity analysis the 7 sub-capabilities that define program Capability #6 were addressed.

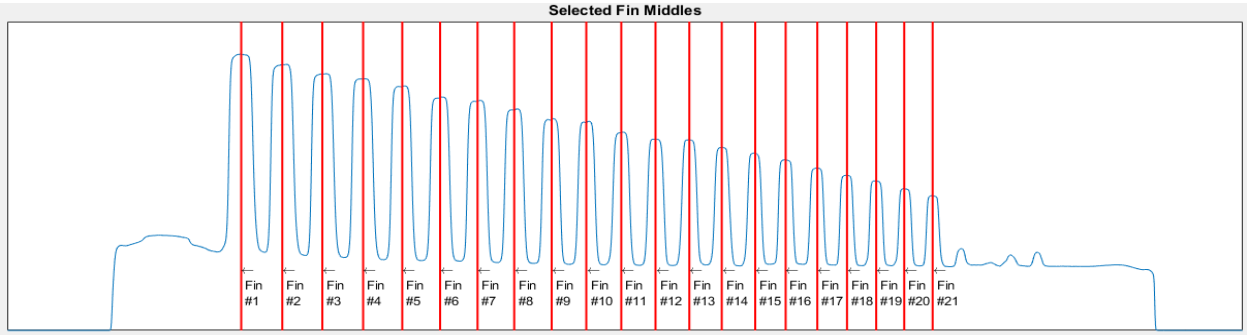


Figure 18: MATLAB returned side-view rendering of fins selected by the user for porosity analysis.

After the image stills corresponding to the center of each fin are selected by the user, which is Capability #5 from Table 3, the program begins to systematically analyze the internal porosity of each fin in the order in which each fin was selected. Each of the 7 sub-capabilities that define required program Capability #6 are executed for every fin selected for analysis.

Capability #6.1 dictates that a region of interest (ROI) must be defined for every fin. It is within the ROI for each fin that all porosity measurements are taken. The ROI of a fin is a scaled-down rectangle based off of the fin's outer boundary rectangle. Using search algorithms designed to detect sharp greyscale pixel transitions the four line segments that comprise the outer boundary rectangle of a fin are detected. Figure 19 illustrates the approximate location and search direction of each search algorithm as well as the respective outer boundary line segment detected by each search algorithm. In Figure 19, search algorithm direction and location are represented by dashed arrows, and detected fin boundaries are represented by solid lines. The specific search algorithm used to detect each fin boundary line segment is indicated by similar color. For example, the pink search algorithm applied to the left side of Figure 19 detects the pink line segment of the outer boundary rectangle.

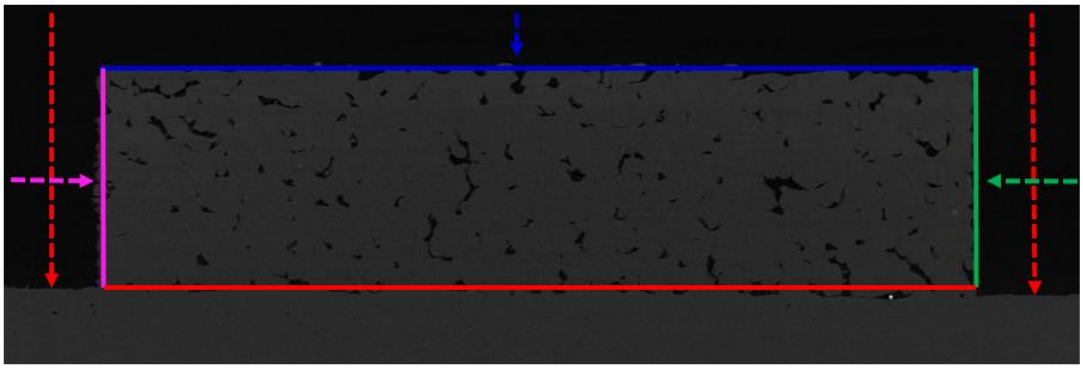


Figure 19: Search algorithms (indicated by dashed arrows) are used to detect the outer boundary (indicated by solid lines) of each fin. Each line segment of the outer boundary rectangle is detected by the search algorithm of the same color.

To reduce the likelihood of incorrectly locating a fin boundary line segment, each search algorithm is applied over a length, and the detected boundary locations are averaged together. The two red search algorithms in Figure 19 detect the top edge of the fin pad, which corresponds

to the bottom horizontal line segment of a fin's outer boundary. After detection of a fin's outer boundary, its ROI is defined.

Through scaling of the fin's outer boundary rectangle, the ROI is defined. Figure 20 illustrates how a fin's outer boundary is scaled-down to form the ROI. The scaling used to generate the ROI is based off of the number of pixels in the length, X, and height, Y, components of a fin's outer boundary rectangle. Length, X, and height, Y, are identified and labeled in Figure 20A. All of the fins analyzed in a fin pad have their X and Y components scaled-down by the same percentage. Calculating porosity statistics using the area inside the ROI, as opposed to the outer boundary rectangle, reduces the likelihood of erroneously identifying dark greyscale pixels trapped between a fin's detected outer boundary rectangle and its actual, nonlinear outer boundary as internal defects. During the process of CT scanning an object its voxel dimensions become known. As a result, when a CT rendering is exported to an image stack, its pixel dimensions become known too. Using the known dimensions of one pixel as well as the known number of pixels contained within a fin's ROI allows program Capability #6.2, which is calculating the ROI area, to be satisfied. To detect instances of porosity within the ROI and satisfy program Capability #6.3 the image segmentation technique known as thresholding was used.

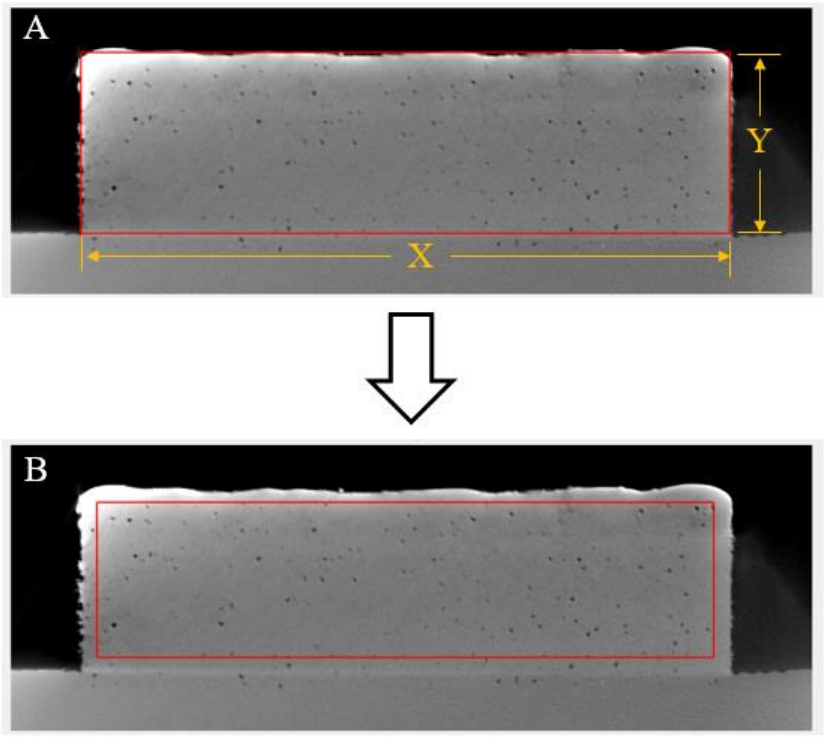


Figure 20: The detected outer boundary rectangle of a fin is reduced by a percentage in its length, X, and height, Y, components to create a ROI rectangle that will be used for porosity measurements.

Internal defects within the ROI appear darker, borderline black, when compared to solid material. Thresholding of the greyscale pixels contained within the ROI allows dark pixels, which correspond to instances of porosity, to be located and identified. Figure 21A is a histogram of greyscale pixels contained within a fin's ROI and Figure 21B is the ROI used to generate the aforementioned histogram. Analysis of numerous ROI pixel histograms led to the selection of a greyscale thresholding value of 100 for analyzing the porosity of all user-selected fins. Additionally, a greyscale thresholding value of 100 yielded the most accurate detection of internal defects within the ROI when visually compared against other thresholding values. In the Figure 21A histogram the selected pixel thresholding value of 100 is tagged with a vertical line. Additionally, the ROI rectangle contained in Figure 21B has undergone defect detection using a

greyscale thresholding value of 100, and detected instances of porosity are highlighted within the ROI. Highlighting detected defects within the ROI satisfies program Capability #6.3.

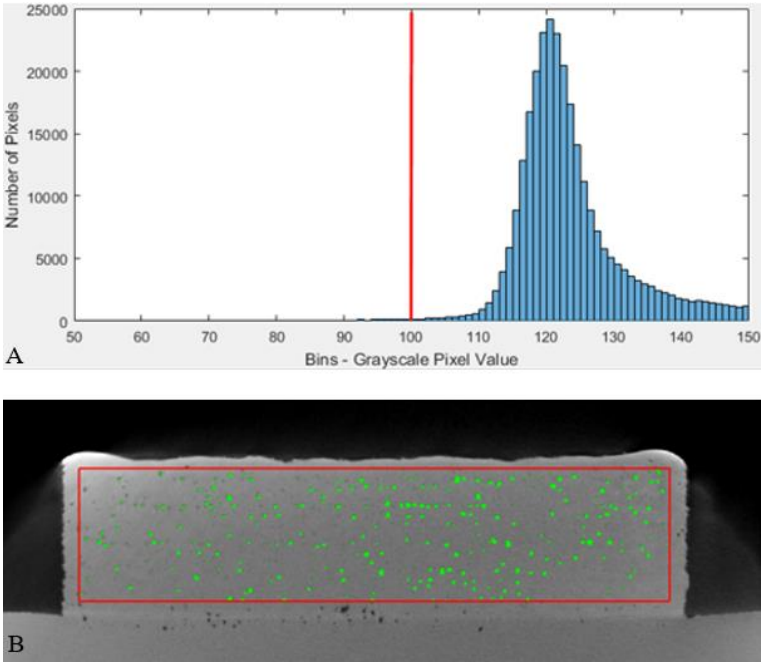


Figure 21: A histogram (A) of greyscale pixel values contained within a ROI (B). Porosity within the ROI (B) was detected using a greyscale thresholding value of 100.

After using thresholding to detect ROI porosity, the following two quantities are calculated: (1) percent porosity within the ROI and (2) total area void of material within the ROI. Calculating percent porosity and total area void of material satisfies program Capabilities #6.4 and #6.5, respectively. The calculated values of percent porosity, total ROI area void of material, and total ROI area are displayed to the user in the manner shown in Figure 22A. Using MATLAB's array manipulation functions, program Capabilities #6.6 and #6.7 of determining the number of ROI defects and generating a histogram of all ROI defects, respectively, were addressed.

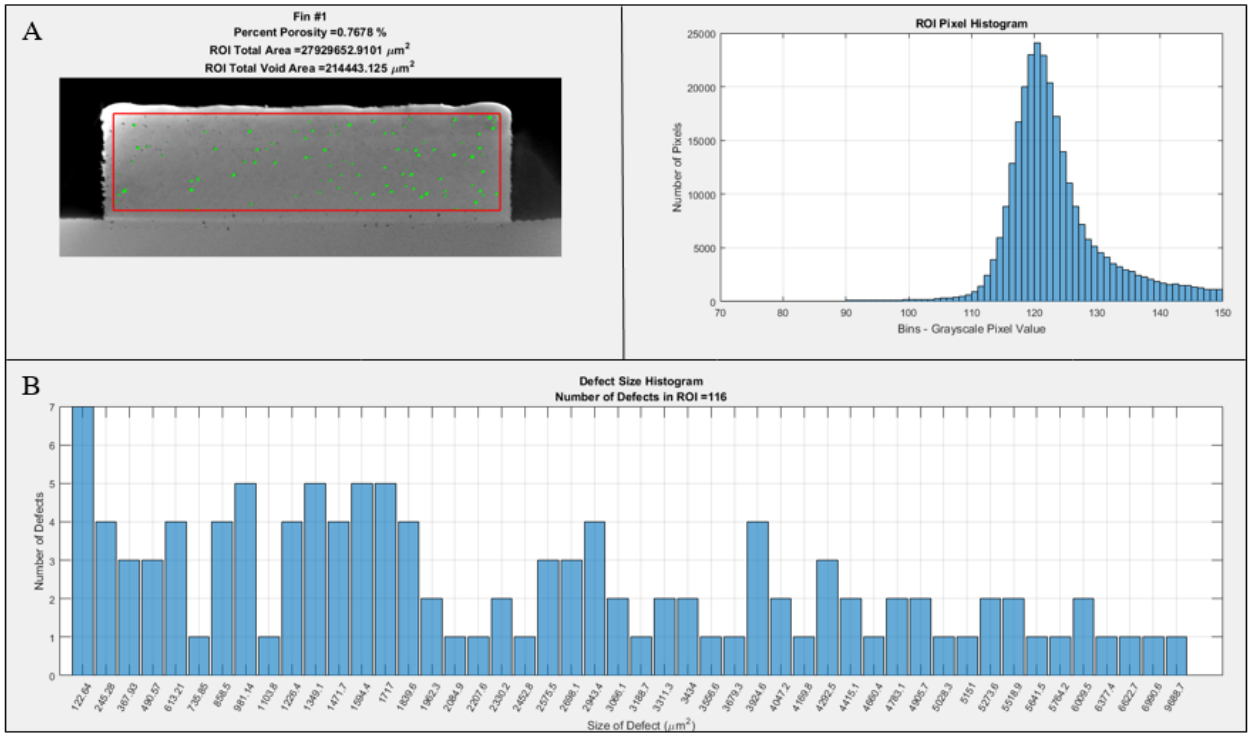


Figure 22: MATLAB returned analysis of porosity for one user-selected fin.

Using the porosity data acquired through thresholding, a histogram documenting the relative size and frequency of all detected defects within the ROI is generated. In the title block of the returned defect size histogram is the number of detected defects within the ROI, as shown in Figure 22B. Generation and display of the defect size histogram and total number of detected ROI defects by the MATLAB program satisfies required Capabilities #6.6 and #6.7 from Table 3. The 7 sub-capabilities that comprise program Capability #6 are executed for each fin selected by the user for analysis. Figure 22 contains a MATLAB window of all porosity and ROI statistics returned to the user for the analysis of one fin.

At the completion of each individual fin analysis, the following pieces of information are compiled into an array: fin number, defect size and frequency data, percent porosity, and ROI area. For each successive fin that is analyzed, the same information is concatenated into an array

with the previous fin's data. When the program is finished analyzing the last fin, the array is exported to a Microsoft Excel file. Exporting defect data to a spreadsheet for all fins analyzed in one user-selection process satisfies program Capability #7. With the 7 required program capabilities outlined in Table 3 met, the developed MATLAB defect detection program was tested to ensure it operates properly.

To verify that the developed MATLAB program performs as intended, custom image stills with known porosity measurements were run through it. The MATLAB code used to generate the custom image stills can be found in Appendix B under the m-file labeled *TestImages*. A detailed explanation of how the custom image stills were used to validate program operation and measurement accuracy can be found in Appendix C. Appendix B also contains the MATLAB code for the defect detection program under the m-file labeled *FinSelectionAnalysis*. After developing and testing the porosity detection program the four fabricated fin pads were analyzed. The porosity data returned by the defect detection program for each fin pad is plotted and discussed in the next chapter.

Chapter 5

Execution of Defect Detection Program and Plotting of Data

After verifying that the defect detection program sequences properly and returns accurate porosity data, the four fabricated fin pads were run through the program and analyzed. Because the four thinnest fins on each fin pad failed to build properly, only Fins #1 through #21 on each pad were analyzed for porosity. Figure 23A - 23D contains the MATLAB returned side-view rendering of each fin pad built at 0°, 30°, 60°, and 90° with respect to the recoater blade, respectively. Under the side-view renderings contained in Figure 23A – 23D are the user-selected fin middles corresponding to each fin selected for analysis using the defect detection program. After executing the program for each fin pad, the returned measurements of percent porosity, total void area, and total build area within the ROI were used to create plots of percent porosity and area fraction.

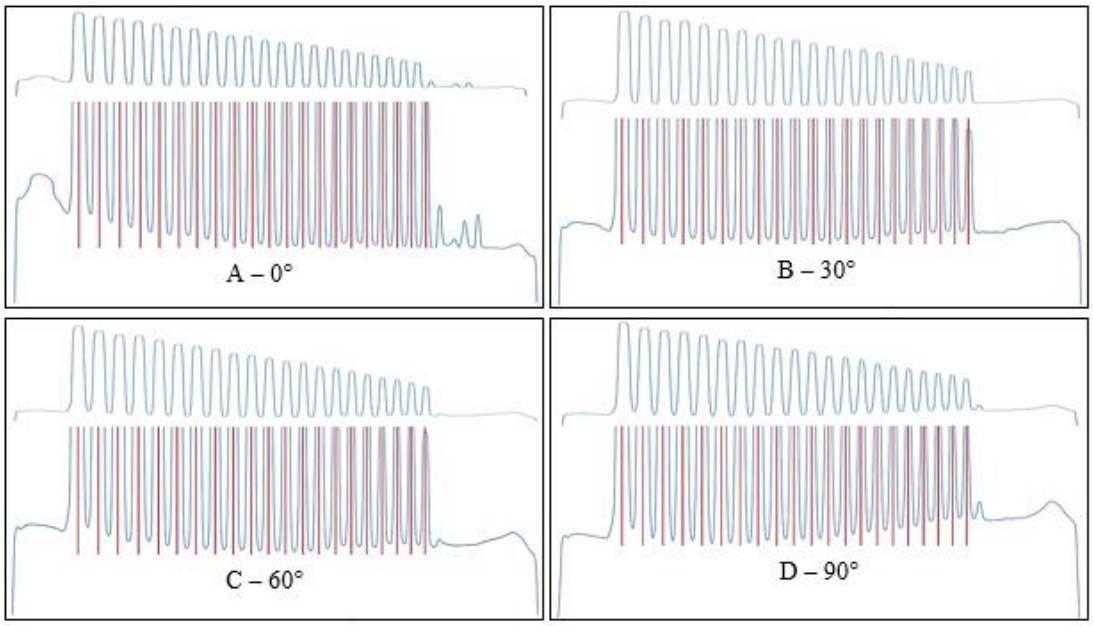
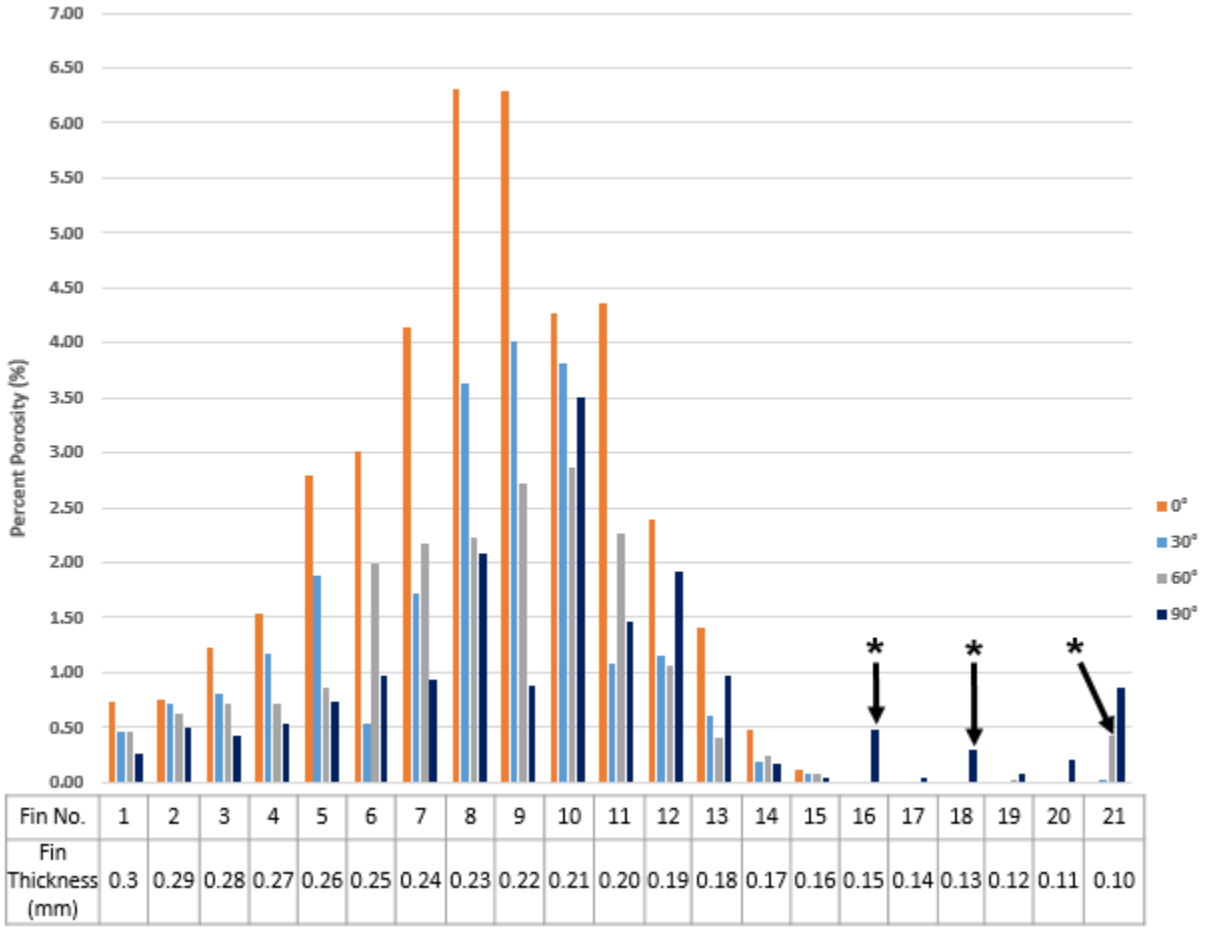


Figure 23: MATLAB returned side-view rendering of fin pads fabricated at 0°, 30°, 60° and 90° with respect to the recoater blade as well as the approximate longitudinal center of each fin selected for analysis on each pad.

Plots of percent porosity and area fraction were created using the returned measurements of percent porosity, ROI total void area, and ROI area for each fin pad. In Figure 24, a plot of percent porosity at angles of 0°, 30°, 60°, and 90° with respect to the recoater blade is displayed for all analyzed fins. Out of the 84 fins that were analyzed, there were only three instances of inaccurate defect detection. The erroneous detection of internal defects caused three porosity values to be inflated. Each inflated percent porosity measurement is identified with an asterisk in Figure 24. The cause of the three erroneous instances of defect detection were greyscale pixels below the thresholding value of 100 trapped between the ROI boundary and the actual, nonlinear fin boundary. A fin still representative of how all three instances of inaccurate defect detection occurred is shown in Figure 25 (note the dark greyscale pixels trapped inside the ROI by the irregular fin boundary along the top). Inspection of the Figure 22 porosity summary returned for

each fin by the MATLAB defect detection and analysis program revealed the instances of inaccurate defect detection.



Fin No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Fin Thickness (mm)	0.3	0.29	0.28	0.27	0.26	0.25	0.24	0.23	0.22	0.21	0.20	0.19	0.18	0.17	0.16	0.15	0.14	0.13	0.12	0.11	0.10

Figure 24: Plot of percent porosity at angles of 0°, 30°, 60° and 90° with respect to the recoater blade for Fin #1 through #21.

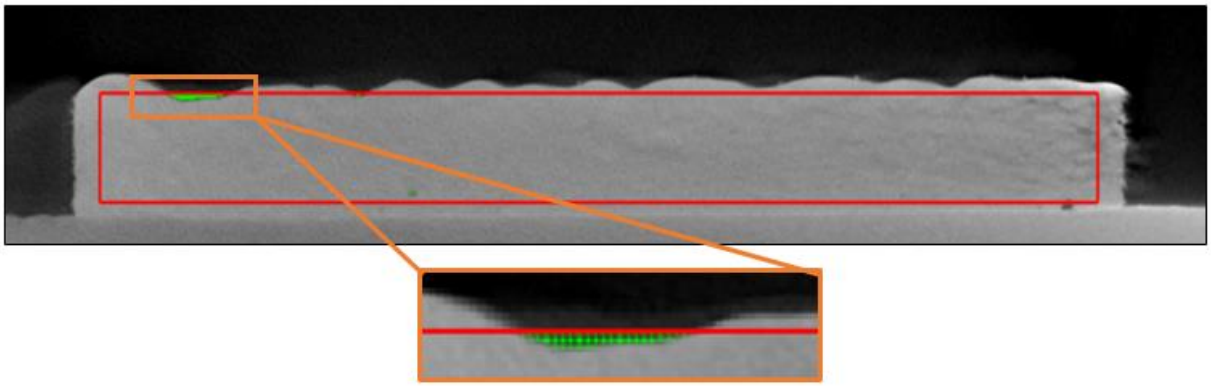
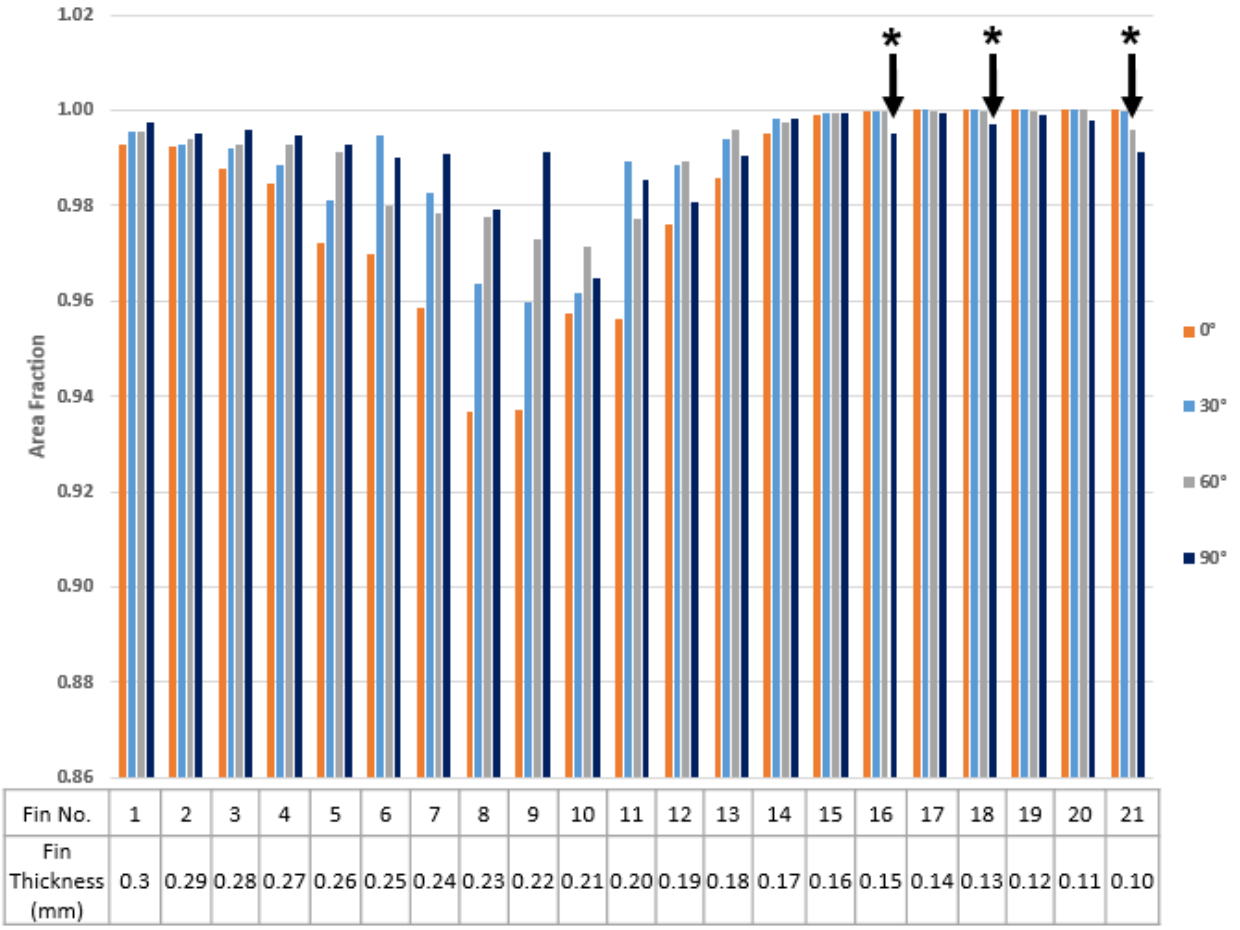


Figure 25: Dark greyscale pixels trapped between the ROI boundary and the actual nonlinear outer boundary result in the erroneous detection of an internal defect.

The other plot created using the returned defect data was area fraction at angles of 0°, 30°, 60°, and 90° with respect to the recoater blade. Figure 26 contains the area fraction plot for all analyzed fins. The asterisks in Figure 26 identify inaccurate area fraction measurements resulting from the three instances of erroneous defect detection. After creating the percent porosity and area fraction plots in Figure 24 and 26, respectively, conclusions were drawn. All drawn conclusions are discussed in the next chapter.



Fin No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Fin Thickness (mm)	0.3	0.29	0.28	0.27	0.26	0.25	0.24	0.23	0.22	0.21	0.20	0.19	0.18	0.17	0.16	0.15	0.14	0.13	0.12	0.11	0.10

Figure 26: Plot of area fraction at angles of 0°, 30°, 60° and 90° with respect to the recoater blade for Fin #1 through #21.

Chapter 6

Data Analysis and Conclusions

The plots generated using the data returned from the defect detection program indicate that part fabrication angle with respect to the recoater blade does have an effect on internal porosity. Inspection of both the percent porosity and area fraction plots from Figure 24 and 26, respectively, show a trend of increasing internal defects when gradually converging to cross-sectional fin thicknesses ranging from 230 microns to 210 microns. Regardless of the fabrication angle with respect to the recoater blade, internal defects in fins near the maximum cross-sectional thickness of 300 microns and minimum cross-sectional thickness of 100 microns show little deviation from each other in terms of internal defects. The described data trend is present in both the area fraction and percent porosity plots for all four fabricated fin pads. The severity of the trend, however, changes with recoater blade angle. Parts fabricated at 0° with respect to the recoater blade exhibit the most severe internal defect trend. Conversely, parts fabricated at 90° with respect to the recoater blade show the least severe internal defect trend. In addition to examining fabrication angle with respect to the recoater blade, laser spot size and corresponding melt pool size were also analyzed for possible links to internal porosity.

The diameter of the laser spot used to fabricate each fin pad was not varied during printing and was measured to be approximately 80 microns. Melt pool diameter, which is closely related to laser spot size, laser intensity and initial substrate conditions (e.g., temperature), was approximated as 110 microns. Fin #1 through #21 on all four fin pads were fabricated using the same laser scan pattern. The width of the inner contour rectangle, however, was adjusted by the

EOS software to accommodate the decreasing thickness of each fin. Figure 27 illustrates how laser spot and melt pool interaction changes with decreasing cross-sectional fin thickness, which results in decreasing width of the inner contour rectangle discussed in Chapter 3. The width of the inner contour rectangle is significant for Fins #14 through #18 because in this fin range both the melt pool and laser beam spot start to overlap. Melt pool overlap occurs in fins less than or equal to 160 microns thick and laser overlap occurs in fins less than or equal to 130 microns thick. Fins less than or equal to 160 microns thick start at Fin #15 on the fin pad and fins less than or equal to 130 microns thick start at Fin #18 on the fin pad.

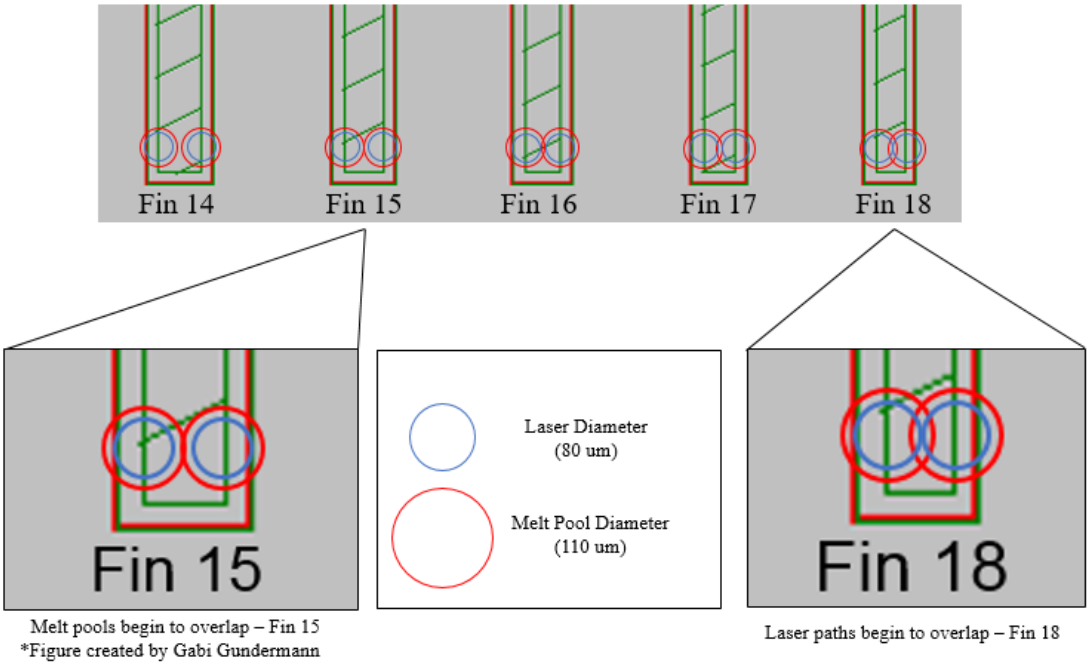


Figure 27: Melt pool and laser spot interaction resulting from the decreasing width of the inner contour rectangle.

The effect of laser spot and melt pool overlap on internal porosity can be seen in Figure 28 and 29. Figure 28 and 29 contain plots of percent porosity and area fraction, respectively, with shaded overlays indicating whether fins were fabricated with melt pool overlap, laser spot

overlap, or with neither laser spot nor melt pool overlap. Inspection of both the percent porosity and area fraction plots contained in Figure 28 and 29 reveal that internal defects in thin-walled structures can be reduced significantly if fabricated using a laser scan pattern and spacing that causes the melt pool of adjacent laser passes to overlap. The melt pool, not the laser spot, is the critical parameter requiring overlap because the melt pool diameter is larger than the laser spot diameter and Figure 28 and 29 indicate that melt pool overlap and laser spot overlap reduce internal porosity equally well. When fabricating PBF thin-walled structures with melt pool overlap, internal defects become independent of build angle with respect to the recoater blade. If there is no melt pool overlap during fabrication internal defects within the structure become dependent on build angle with respect to the recoater blade. When no melt poor overlap occurs, internal porosity is most significant in parts fabricated at 0° with respect to the recoater blade and least significant in parts fabricated at 90° with respect to the recoater blade.

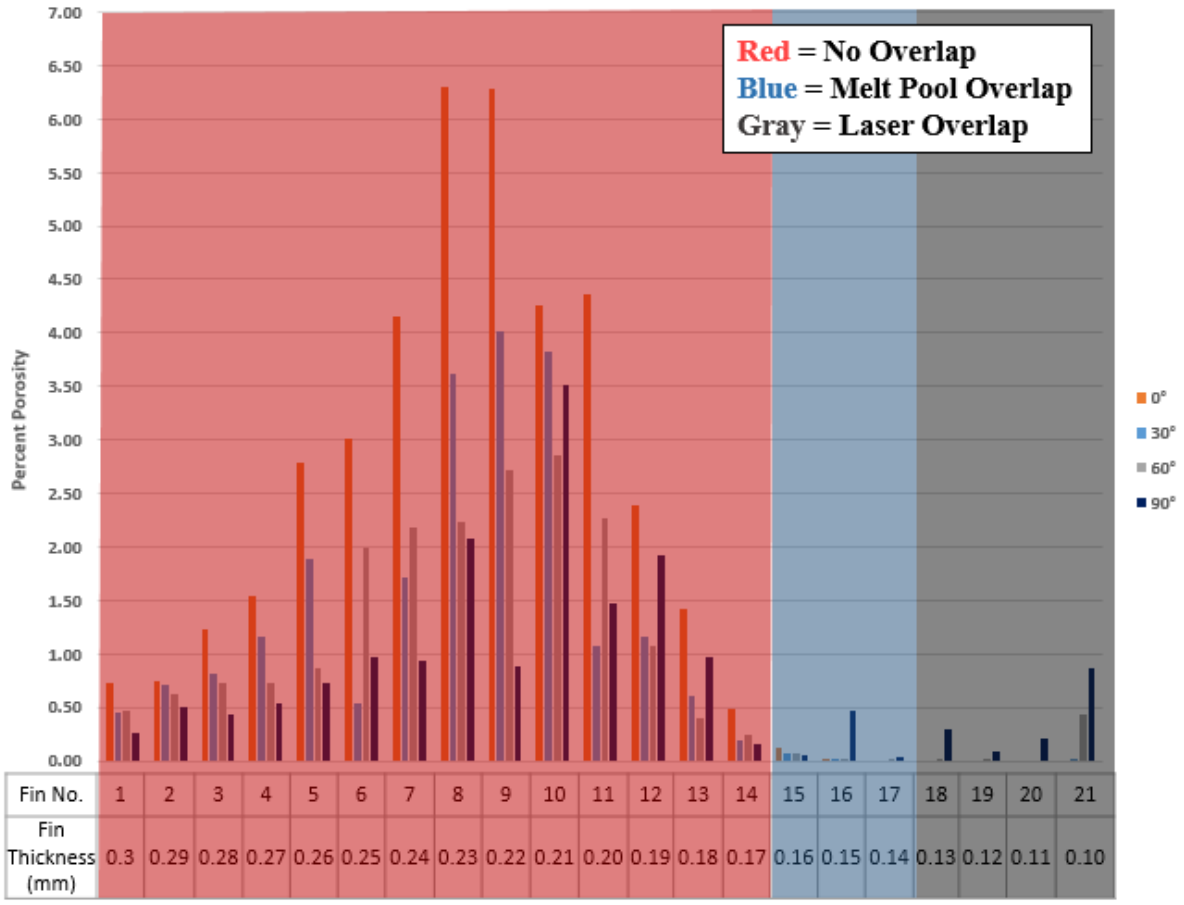


Figure 28: Plot of percent porosity at angles of 0°, 30°, 60° and 90° with respect to the recoater blade for Fin #1 through #21. An overlay indicating which fins were fabricated with laser spot overlap, melt pool overlap and neither laser spot nor melt pool overlap is included in the plot.

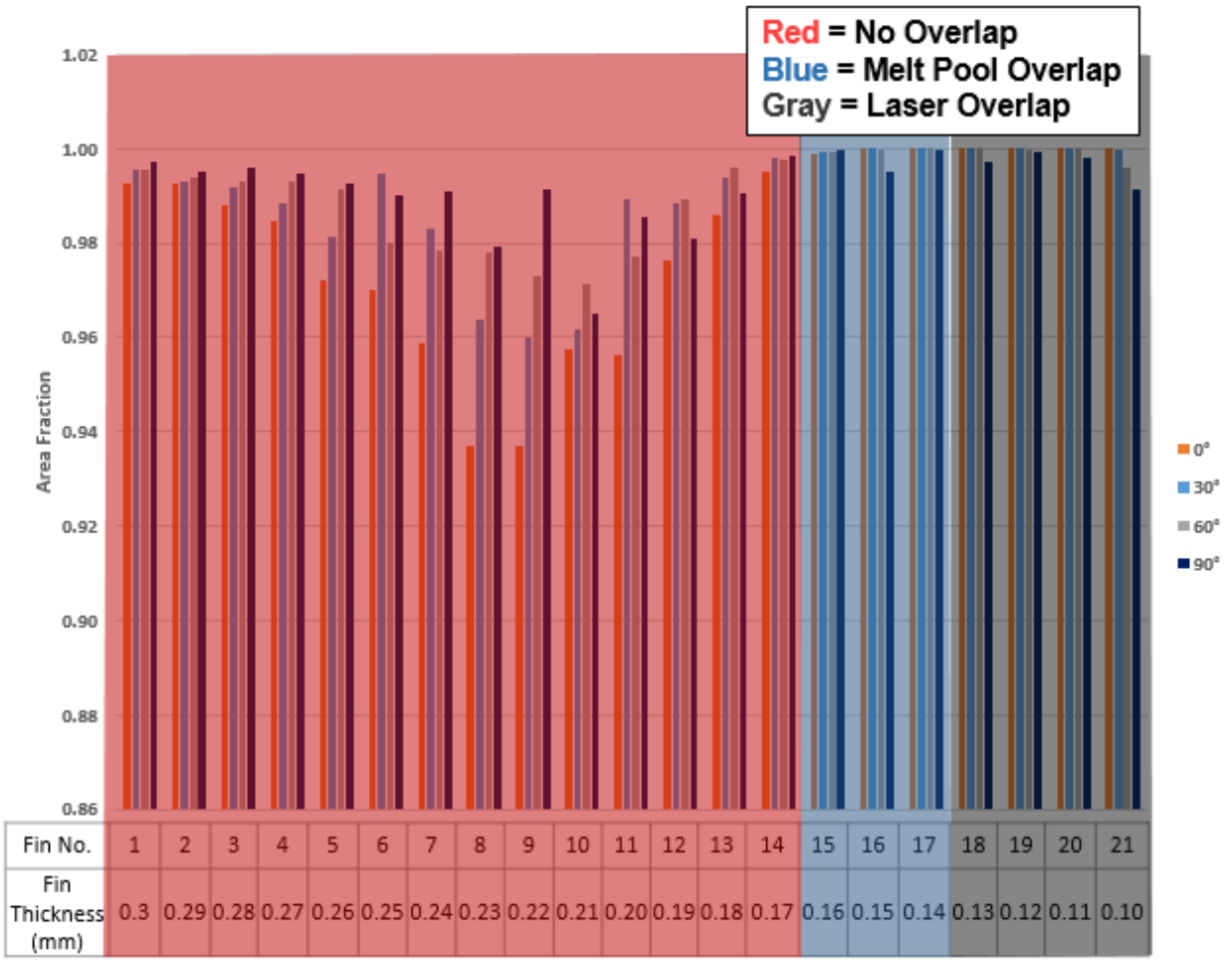


Figure 29: Plot of area fraction at angles of 0°, 30°, 60° and 90° with respect to the recoater blade for Fin #1 through #21. An overlay indicating which fins were fabricated with laser spot overlap, melt pool overlap and neither laser spot nor melt pool overlap is included in the plot.

In conclusion, the MATLAB defect detection program performed well in its task of analyzing four fin pad image stacks for internal porosity. The data returned by the program was meaningful and allowed for the creation of plots showing a clear relationship between internal porosity and fabrication angle with respect to the recoater blade. Through additional analysis of laser spot diameter, melt pool diameter and laser scan path the effect of fabrication angle with respect to the recoater blade on internal porosity was further revealed. The next chapter discusses limitations of the MATLAB porosity detection and analysis program as well as future work.

Chapter 7

Limitations of MATLAB Defect Detection Program and Future Work

A noteworthy limitation of the MATLAB defect detection program is that it analyzes porosity two-dimensionally. Analyzing porosity without considering its depth component can inaccurately portray a structure's true porosity because small defects that extend through multiple image stills are not considered equal to large defects that are centralized to one or two image stills. Additional limitations of the defect detection program are the rigid design constraints imposed on any thin-walled structure that is to be analyzed. Thin-walled structures must be rectangular and can have no internal geometry; otherwise, the program might define the structure's ROI incorrectly or mistake internal geometry as porosity. Another program limitation is that only thin-walled structures can be reliably analyzed for porosity.

The inability to accurately analyze the porosity of structures except for those that are thin stems from the requirement that users approximate the middle of each walled structure. Because porosity measurements are calculated using the image still corresponding to the user-selected middle of a walled structure, the accuracy of any returned porosity measurements becomes more questionable as thickness increases. Porosity measurements from multiple image stills within a walled structure can be averaged together, however, to improve the accuracy of porosity data returned for both thin and thick-walled structures. Currently, the MATLAB defect detection program is not designed to average together multiple porosity measurements from a single walled structure.

Future work includes attempting to replicate the porosity data from the four Ti-6Al-4V fin pads fabricated on the EOSINT M280 PBF printer using four aluminum fin pads fabricated on a ProX Direct Metal Printing (DPM) 320 PBF printer. Again, laser spot diameter, melt pool diameter, laser scan pattern, and fabrication angle with respect to the recoater blade will be analyzed.

Bibliography

Atkinson, H. V., and S. Davies. "Fundamental Aspects of Hot Isostatic Pressing: An Overview." *Metallurgical and Materials Transactions A* 31.12 (2000): 2981-3000. Web. 1 Apr. 2017.

General Electric Company. *GE Inspection Technologies: phoenix v|tome|x m* . N.p.: General Electric Company, 2016. Print.

Gong, Haijun, Khalid Rafi, Hengfeng Gu, Thomas Starr, and Brent Stucker. "Analysis of Defect Generation in Ti–6Al–4V Parts Made Using Powder Bed Fusion Additive Manufacturing Processes." *Additive Manufacturing* 1-4 (2014): 87-98. Web. 10 Apr. 2017.

King, W. E., A. T. Anderson, R. M. Ferencz, N. E. Hodge, C. Kamath, S. A. Khairallah, and A. M. Rubenchik. "Laser Powder Bed Fusion Additive Manufacturing of Metals; Physics, Computational, and Materials Challenges." *Applied Physics Reviews* 2.4 (2015): 041304. Web. 5 Apr. 2017.

Li, Jing , Connor Myant, and Billy Wu. *The Current Landscape for Additive Manufacturing Research*. Rep. N.p.: Imperial College Additive Manufacturing Network, 2016. Print.

Lu, S. L., H. P. Tang, Y. P. Ning, N. Liu, D. H. Stjohn, and M. Qian. "Microstructure and Mechanical Properties of Long Ti-6Al-4V Rods Additively Manufactured by Selective Electron Beam Melting Out of a Deep Powder Bed and the Effect of Subsequent Hot Isostatic Pressing." *Metallurgical and Materials Transactions A* 46.9 (2015): 3824-834. Web. 10 Apr. 2017.

Mercelis, Peter, and Jean-Pierre Kruth. "Residual stresses in selective laser sintering and selective laser melting." *Rapid Prototyping Journal* 12.5 (2006): 254-65. Web. 31 Mar. 2017.

Noel, Julien. "Advantages of CT in 3D Scanning of Industrial Parts." *3D Scanning Technologies Magazine* Dec. 2008: 18-23. Print.

Simpson, Timothy W. "AM Needs MEs: 3-D Printing of Metals is a Process Full of Unknowns; Mechanical Engineers Can Fill in Many of the Gaps. ." *Mechanical Engineering* Aug. 2015: 30-35. Print.

Tang, Ming, P. Chris Pistorius, and Jack L. Beuth. "Prediction of Lack-of-Fusion Porosity for Powder Bed Fusion." *Additive Manufacturing* 14 (2017): 39-48. Web. 1 Apr. 2017.

Withers, P. J. "Residual Stress and its Role in Failure." *Reports on Progress in Physics* 70.12 (2007): 2211-264. Web. 31 Mar. 2017.

Yadroitsev, I., P. Krakhmalev, and I. Yadroitsava. "Hierarchical Design Principles of Selective Laser Melting for High Quality Metallic Objects." *Additive Manufacturing* 7 (2015): 45-56. *Elsevier*. Web. 27 Feb. 2017.

Yap, C. Y., C. K. Chua, Z. L. Dong, Z. H. Liu, D. Q. Zhang, L. E. Loh, and S. L. Sing. "Review of Selective Laser Melting: Materials and Applications." *Applied Science Reviews* 2.4 (2015): n. pag. Web. 8 Feb. 2017.

Appendix A

Fin Pad Image Stacks

The image stacks for each analyzed fin pad can be viewed and/or downloaded by following the hyperlinks listed in Table A1.

Table A1: Hyperlinks that route to the image stacks of each analyzed fin pad.

Fin Pad - 0° w.r.t Recoater Blade
Fin Pad - 30° w.r.t Recoater Blade
Fin Pad - 60° w.r.t Recoater Blade
Fin Pad - 90° w.r.t Recoater Blade

Appendix B

MATLAB Code

Code B1: *TestImages.m*

```
clear, clc

A = ones(150,450)*160;

B = ones(150,100)*160;

C = ones(50,450)*160;

D = zeros(50,100)*160;

E = horzcat(A,B,A);

F = horzcat(C,D,C);

G = vertcat(E,F,E);

H = zeros(150,1500);

I = zeros(350, 250);

J = ones(300,1500)*160;

K = horzcat(I,G,I);

test_matrix2 = vertcat(H,K,J);

test_matrix2 = mat2gray(test_matrix2, [0 255]);

imwrite(test_matrix2,'C:\Users\Matt\Pictures\FinTestImages\TestImage2.jpg')

AA = ones(160,450)*160;

BB = ones(160,100)*160;

CC = ones(20,450)*160;

DD = zeros(20,100)*160;
```

```
EE = horzcat(AA,BB,AA);

FF = horzcat(CC,DD,CC);

GG = vertcat(EE,FF,EE);

HH = zeros(150,1500);

II = zeros(340, 250);

JJ = ones(300,1500)*160;

KK = horzcat(II,GG,II);

test_matrix4 = vertcat(HH,KK,JJ);

test_matrix4 = mat2gray(test_matrix4, [0 255]);

imwrite(test_matrix4,'C:\Users\Matt\Pictures\FinTestImages\TestImage4.jpg')
```

Code B2: *FinSelectionAnalysis.m*

```
clc,clear

Total_Number_Of_Images = ; %Input number of images in image stack. See
example below.

%Total_Number_Of_Images = 2355;

image_sum_total = zeros(1,Total_Number_Of_Images);

for P = 1:Total_Number_Of_Images;

image_name = ; %Input file path and name. See example below.

%image_name = (strcat('E:\Cropped MATLAB Image Stack\Fin90
(',num2str(P),').tif'));

image_matrix = imread(image_name);

image_matrix = rgb2gray(image_matrix);

image_matrix_greater_than = (image_matrix > 60);

image_sum_vector = sum(image_matrix_greater_than);
```

```
image_sum_total(P) = sum(image_sum_vector);

end

image_number = size(image_sum_total,2);

image_number = 1:image_number;

plot(image_number, image_sum_total)

title('Select the Middle of Each Fin')

set(gca,'xtick',[],'ytick',[])

[x,y] = ginput;

images_numbers = round(x,0);

images_numbers_size = size(images_numbers,1);

images_numbers = reshape(images_numbers,[1,images_numbers_size]);

hold on

y_axis_max = ylim;

y_axis_max = y_axis_max(2);

title('Selected Fin Middles')

for J = 1:images_numbers_size;

    x = [images_numbers(J) images_numbers(J)];
    y = [0 y_axis_max];
    plot(x,y,'Color','r','LineStyle','-','linewidth',1.5), grid
    x = images_numbers(J);
    y = (y_axis_max*.15);
    str = (num2str(J));
    str = strcat(' #', str );
    str = strcat({'\leftarrow', ' Fin', str});
    text(x,y,str)

end

hold off

T = size(images_numbers,2);

Pixel_Row_Middle_Average = zeros(1,T);

Pixel_Row_Average_Left_Right_Average = zeros(1,T);
```

```
Pixel_Column_Left_Comp = zeros(1,T);

Pixel_Column_Right_Comp = zeros(1,T);

Pixel_Column_Difference_All = zeros(1,T);

for Q = 1:T

    input_image_name = ; %Input file path and name. See example below.

    %input_image_name = (strcat('E:\Cropped MATLAB Image Stack\Fin90
('',num2str(images_numbers(Q)),').tif'));

    input_image_changed = imread(input_image_name);

    input_image_changed = rgb2gray(input_image_changed);

    M = size(input_image_changed,1);

    N = size(input_image_changed,2);

for k=1:N
    Column_N_of_two_d_image = input_image_changed(:,k);
    for n= 1:(M-2)
        if Column_N_of_two_d_image(n)>100 && Column_N_of_two_d_image(n+1)>100
&& Column_N_of_two_d_image(n+2)>100
            Pixel_Row(k)=n;
            break
        end

    end
end

Pixel_Row_Left = Pixel_Row(1:(round((N *0.05),0)));

Pixel_Row_Middle = Pixel_Row((round((N *0.1),0)):(round((N *0.9),0)));

Pixel_Row_Right = Pixel_Row((round((N *0.95),0)):N);

Pixel_Row_Middle_Average (Q) = round(mean(Pixel_Row_Middle),0);

Pixel_Row_Average_Left_Right_Average(Q) = -1 + round(mean([Pixel_Row_Left,
Pixel_Row_Right]),0);

Two_d_Image_Transposed = (input_image_changed.');

for s=1:M

Columns_Two_d_Image_Trans = Two_d_Image_Transposed(:,s);
```

```
    for q = 1:(round((N *0.5),0));
        if  Columns_Two_d_Image_Trans(q)>100 &&
Columns_Two_d_Image_Trans(q+1)>100 && Columns_Two_d_Image_Trans(q+2)>100
            Pixel_Column_Left(s)=q;
            break
        end

    end

 Columns_Two_d_Image_Trans = flipud(Columns_Two_d_Image_Trans);

    for w = 1:(round((N *0.5),0));
        if  Columns_Two_d_Image_Trans(w)>100 &&
Columns_Two_d_Image_Trans(w+1)>100 && Columns_Two_d_Image_Trans(w+2)>100
            Pixel_Column_Right(s)=w;
            break
        end

    end

end

Pixel_Column_Left = Pixel_Column_Left(Pixel_Row_Middle_Average(Q) :
Pixel_Row_Average_Left_Right_Average(Q));

Pixel_Column_Left_Comp(Q) = mode(Pixel_Column_Left);

Pixel_Column_Right = Pixel_Column_Right(Pixel_Row_Middle_Average(Q) :
Pixel_Row_Average_Left_Right_Average(Q));

Pixel_Column_Right_Comp(Q) =  N + 1 - mode(Pixel_Column_Right);

Pixel_Column_Difference_All(Q) = Pixel_Column_Right_Comp(Q) -
Pixel_Column_Left_Comp(Q);

end

Pixel_Column_Difference = 1 + round(mean(Pixel_Column_Difference_All),0);

Pixel_xy_micrometer = 14;

Pixel_Threshold = 100;

Per_Por = zeros(1,T);

ROI_Area = zeros(1,T);

for R = 1:T

input_image_name = ; %Input file path and name. See example below.
```

```
%input_image_name = (strcat('E:\Cropped MATLAB Image Stack\Fin90
(',num2str(images_numbers(R)),').tif'));

input_image_unchanged = imread(input_image_name);

input_image_changed = imread(input_image_name);

input_image_changed = rgb2gray(input_image_changed);

input_image_unchanged = rgb2gray(input_image_unchanged);

Pixel_Row = round((Pixel_Row_Average_Left_Right_Average(R) -
Pixel_Row_Middle_Average(R) + 1) * 0.07 , 0);

Pixel_Row_Middle_Average_ROI = (Pixel_Row_Middle_Average(R) + Pixel_Row);

Pixel_Row_Average_Left_Right_Average_ROI =
(Pixel_Row_Average_Left_Right_Average(R) - Pixel_Row);

Pixel_Column = round((Pixel_Column_Right_Comp(R) - Pixel_Column_Left_Comp(R)
+ 1) * 0.025 , 0);

Pixel_Column_Left_ROI = (Pixel_Column_Left_Comp(R) + Pixel_Column);

Pixel_Column_Right_ROI = (Pixel_Column_Right_Comp(R) - Pixel_Column);

Region_of_Interest = input_image_unchanged(Pixel_Row_Middle_Average_ROI :
Pixel_Row_Average_Left_Right_Average_ROI , Pixel_Column_Left_ROI :
Pixel_Column_Right_ROI); % Defining the ROI box

Total_Pixels_in_ROI = (size(Region_of_Interest,1)) *
(size(Region_of_Interest,2));

Region_of_Interest_greater_than = (Region_of_Interest < Pixel_Threshold );

Region_of_Interest_sum_vector = sum(Region_of_Interest_greater_than);

Region_of_Interest_sum_total = sum(Region_of_Interest_sum_vector);

Percent_Porosity_Number = ((Region_of_Interest_sum_total /
Total_Pixels_in_ROI) * 100);

Per_Por(R) = Percent_Porosity_Number;

ROI_Area(R) = Pixel_xy_micrometer * Pixel_xy_micrometer *
Total_Pixels_in_ROI;

[row,col] = find(Region_of_Interest < Pixel_Threshold);

if Percent_Porosity_Number == 0
    subplot1 = 1;
    subplot2 = 1;
```

```
    subplot3 = 1;
    subplot4 = 1;
    Plot_Open = 0;
else
    Plot_Open = 1;
    subplot1 = 2;
    subplot2 = 2;
    subplot3 = 1;
    subplot4 = 2;
    subplot5 = 3;
    subplot6 = 4;
end

figure('Name',input_image_name)

set(gcf, 'Position', get(0, 'Screensize')),imshow(input_image_changed)

subplot(subplot1,subplot2,subplot4)

histogram(Region_of_Interest) , grid on

xlabel('Bins - Grayscale Pixel Value')

ylabel('Number of Pixels')

title('ROI Pixel Histogram')

xlim([70 150])

ylim([0 25000])

set(gca,'YTickLabel',num2str(get(gca,'YTick').'))

subplot(subplot1,subplot2,subplot3), imshow(input_image_changed)

Selected_Fin_Number = (strcat('Fin #', num2str(R)));

Percent_Porosity_String = strcat('Percent Porosity = ' ,
num2str(Percent_Porosity_Number) ,' %');

ROI_Total_Area = Pixel_xy_micrometer * Pixel_xy_micrometer *
Total_Pixels_in_ROI;

ROI_Defect_Area = (Percent_Porosity_Number * (1/100) * ROI_Total_Area);

ROI_Total_Area_String = strcat('ROI Total Area = ' , num2str(ROI_Total_Area)
,' \mum^2');

ROI_Defect_Area_String = strcat('ROI Total Void Area = ' ,
num2str(ROI_Defect_Area) ,' \mum^2');
```

```
title({Selected_Fin_Number ; Percent_Porosity_String; ROI_Total_Area_String;
ROI_Defect_Area_String })

hold on

    scatter(col+Pixel_Column_Left_ROI, row+Pixel_Row_Middle_Average_ROI,1,
'filled', 's', 'g')

    x = [Pixel_Column_Left_ROI Pixel_Column_Right_ROI];
    y = [Pixel_Row_Average_Left_Right_Average_ROI
Pixel_Row_Average_Left_Right_Average_ROI];
    plot(x,y,'Color','r','LineStyle','-','linewidth',1.5)

    x = [Pixel_Column_Left_ROI Pixel_Column_Right_ROI];
    y = [Pixel_Row_Middle_Average_ROI Pixel_Row_Middle_Average_ROI];
    plot(x,y,'Color','r','LineStyle','-','linewidth',1.5)

    x = [Pixel_Column_Left_ROI Pixel_Column_Left_ROI];
    y = [Pixel_Row_Average_Left_Right_Average_ROI
Pixel_Row_Middle_Average_ROI];
    plot(x,y,'Color','r','LineStyle','-','linewidth',1.5);

    x = [Pixel_Column_Right_ROI Pixel_Column_Right_ROI];
    y = [Pixel_Row_Average_Left_Right_Average_ROI
Pixel_Row_Middle_Average_ROI];
    plot(x,y,'Color','r','LineStyle','-','linewidth',1.5);

hold off

if Plot_Open == 1;

[Defect_Matrix,Number_of_Defects] = bwlabel(Region_of_Interest_greater_than);

Void_Pixels = zeros(1,Number_of_Defects);

for V = 1:Number_of_Defects;

Z = find(Defect_Matrix==V);

Void_Pixels(V) = size(Z,1);

end

Void_Pixels_Integer = find(Void_Pixels == mode(Void_Pixels));

Void_Pixels_Integer = size(Void_Pixels_Integer,2);

Void_Area = (Void_Pixels * Pixel_xy_micrometer * Pixel_xy_micrometer);

subplot(subplot1,subplot2,subplot5:subplot6)
```

```
ROI_Defect_Number = strcat('Number of Defects in ROI = ' ,
num2str(Number_of_Defects));

histogram(categorical(Void_Area))

title({'Defect Size Histogram'; ROI_Defect_Number })

grid on

set(gca,'ytick',0:Void_Pixels_Integer);

ylabel('Number of Defects')

xlabel('Size of Defect (\mum^2)')

Void_Area = sort(Void_Area);

else

    Void_Area = 0;

end

VOID_Matrix_Size = zeros(1,3);

KeyXYZ = 1;

XYZ = R;

    VOID_Matrix_Size(XYZ) = size(Void_Area, 2);

    if XYZ == 1;

    else
        KeyXYZ = 2;
        if VOID_Matrix_Size(XYZ) == Matrix_Compile_Columns;
            Matrix_Running = vertcat(Matrix_Compile, Void_Area);
        elseif VOID_Matrix_Size(XYZ) > Matrix_Compile_Columns;
            Zero_Difference = (VOID_Matrix_Size(XYZ) -
Matrix_Compile_Columns);
            Matrix_Compile_Rows = size(Matrix_Compile, 1);
            Zero_Matrix = zeros(Matrix_Compile_Rows,Zero_Difference);
            Matrix_Running = horzcat(Matrix_Compile,Zero_Matrix);
            Matrix_Running = vertcat(Matrix_Running , Void_Area);
        else
            Zero_Difference = (Matrix_Compile_Columns -
VOID_Matrix_Size(XYZ));
            Zero_Matrix = zeros(1,Zero_Difference);
            Matrix_Running = horzcat(Void_Area,Zero_Matrix);
            Matrix_Running = vertcat(Matrix_Compile, Matrix_Running);
        end
```

```
    end

    if KeyXYZ == 1;
        Matrix_Compile = Void_Area;
    else
        Matrix_Compile = Matrix_Running;
    end

    Matrix_Compile_Columns = size(Matrix_Compile,2);

end

Fin_Numbers = (1:T)';

Matrix_Compile = horzcat(Fin_Numbers,Per_Por',ROI_Area',
round(Matrix_Compile,2));

filename = ; %Input file path and name. See example below.

%filename = 'E:\Cropped MATLAB Image Stack\PorosityData.xlsx';

xlswrite(filename,Matrix_Compile')
```

Appendix C

Validation of Defect Detection Program

To verify the accuracy of all ROI measurements returned to the user, two custom-designed test images with known porosity measurements were developed and run through the defect detection program. Each test image was designed in MATLAB such that there is only one defect within the ROI. Figure C1 and C2 show Test Image #1 and Test Image #2, respectively. Test Image #1 and #2 were designed to replicate two different user-selected fin middles. Replicating two different fin middles means the width of each fin should be similar and the height of each fin should be dissimilar. To satisfy the requirement of dissimilar height the fin in Test Image #2 was designed 10 pixels shorter than the fin in Test Image #1. Table C1 lists the columns and rows corresponding to the fin boundary in Test Image #1. The top left corner of each test image is the origin and all row and column counts are positive integers. Table C1 also lists the columns and rows corresponding to the ROI for Test Image #1. All intermediate calculations used to determine the ROI are included in Table C1. Scaling percentages of 2.5% for the width and 7.0% for the height were used when determining the ROI. All information outlined in Table C1 for Test Image #1 is updated for Test Image #2 in Table C2.

Figure C1: Test Image #1.

Figure C2: Test Image #2.

Table C1: Fin and ROI boundary locations for Test Image #1.

-	Left Boundary	Right Boundary	Top Boundary	Bottom Boundary
Fin	Pixel Column @ 251	Pixel Column @ 1250	Pixel Row @ 151	Pixel Row @ 500
Calculations: Column difference = 1250 pixels – 251 pixels + 1 pixel = 1000 pixels Row difference = 500 pixels – 151 pixels + 1 pixel = 350 pixels ROI columns @ 251 pixels + (1000 pixels x 0.025) & 1250 pixels - (1000 pixels x 0.025) ROI columns @ 276 pixels & 1225 pixels ROI rows @ 151 pixels + (350 pixels x 0.07) & 500 pixels - (350 pixels x 0.07) ROI rows @ 176* pixels & 475* pixels				
ROI	Pixel Column @ 276	Pixel Column @ 1225	Pixel Row @ 176*	Pixel Row @ 475*

***Calculation rounded to nearest integer.**

Table C2: Fin and ROI boundary locations for Test Image #2.

-	Left Boundary	Right Boundary	Top Boundary	Bottom Boundary
Fin	Pixel Column @ 251	Pixel Column @ 1250	Pixel Row @ 151	Pixel Row @ 490
Calculations: Column difference = 1250 pixels – 251 pixels + 1 pixel = 1000 pixels Row difference = 490 pixels – 151 pixels + 1 pixel = 340 pixels ROI columns @ 251 pixels + (1000 pixels x 0.025) & 1250 pixels - (1000 pixels x 0.025) ROI columns @ 276 pixels & 1225 pixels ROI rows @ 151 pixels + (340 pixels x 0.07) & 490 pixels - (340 pixels x 0.07) ROI rows @ 175* pixels & 466* pixels				
ROI	Pixel Column @ 276	Pixel Column @ 1225	Pixel Row @ 175*	Pixel Row @ 466*

***Calculation rounded to nearest integer.**

After determining the specific columns and rows that define the ROI for Test Image #1 and #2, values for percent porosity, total ROI area and total ROI void area were manually calculated. All intermediate calculations used to determine percent porosity, total ROI area and total ROI void area for Test Image #1 and #2 are shown in Table C3 and C4, respectively. A pixel with dimensions of 11.0744 um by 11.0744 um was used for all area dependent calculations. The MATLAB returned porosity analyses for Test Image #1 and #2 are shown in Figure C3 and C4, respectively.

Table C3: Percent porosity, total ROI area and total ROI void area calculations for Test Image #1. Necessary intermediate calculations are shown.

Calculations: Total pixel elements below a grayscale threshold value of 100 = 50 pixels x 100 pixels = 5000 $pixels^2$	
ROI Total Void Area ($pixels^2$)	5000
Calculations: 5000 $pixels^2$ x 11.0744 um x 11.0744 um = 613211.7 um^2	
ROI Total Void Area (um^2)	613211.7
Calculations: ROI column difference = 1225 pixels - 276 pixels + 1 pixel = 950 pixels ROI row difference = 475 pixels -176 pixels + 1 pixel = 300 pixels ROI total pixel elements = 950 pixels x 300 pixels = 285000 $pixels^2$	
ROI Total Area ($pixels^2$)	285000
Calculations: 285000 $pixels^2$ x 11.0744 um x 11.0744 um = 34953065.6 um^2	
ROI Total Area (um^2)	34953065.6
Calculations: Percent porosity = 100 x (613211.7 um^2 / 34953065.6 um^2) = 1.7544 %	
Percent Porosity (%)	1.7544

Table C4: Percent porosity, total ROI area and total ROI void area calculations for Test Image #2. Necessary intermediate calculations are shown.

Calculations: Total pixel elements below a grayscale threshold value of 100 = 20 pixels x 100 pixels = 2000 $pixels^2$	
ROI Total Void Area ($pixels^2$)	2000
Calculations: 2000 $pixels^2$ x 11.0744 um x 11.0744 um = 245284.7 um^2	
ROI Total Void Area (um^2)	245284.7
Calculations: ROI column difference = 1225 pixels - 276 pixels + 1 pixel = 950 pixels ROI row difference = 466 pixels - 175 pixels + 1 pixel = 292 pixels ROI total pixel elements = 950 pixels x 292 pixels = 277400 $pixels^2$	
ROI Total Area ($pixels^2$)	277400
Calculations: 277400 $pixels^2$ x 11.0744 um x 11.0744 um = 34020983.8 um^2	
ROI Total Area (um^2)	34020983.8
Calculations: Percent porosity = 100 x (245284.7 um^2 / 34020983.8 um^2) = 0.72098 %	
Percent Porosity (%)	0.72098

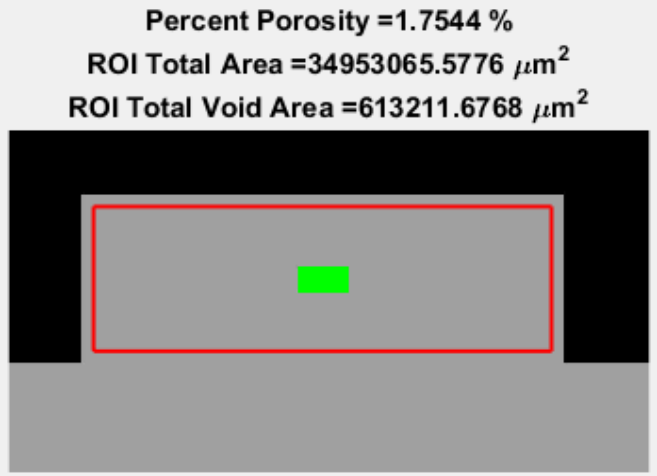


Figure C3: MATLAB returned porosity analysis for Test Image #1.

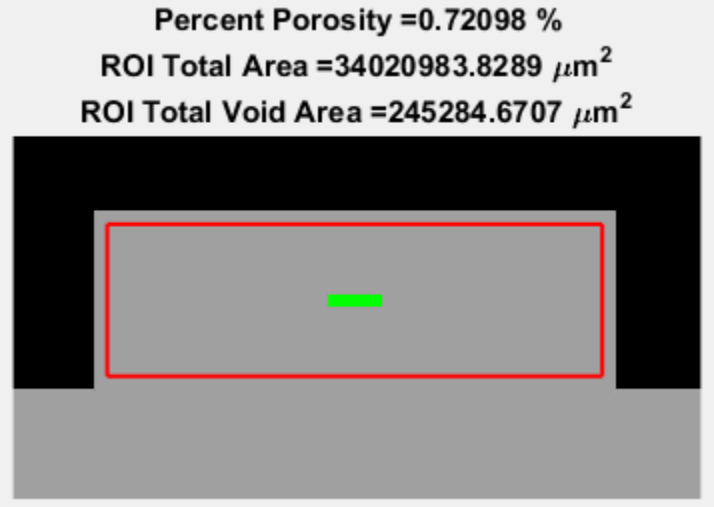


Figure C4: MATLAB returned porosity analysis for Test Image #2.

After manually calculating percent porosity, total ROI area and total ROI void area for Test Image #1 and #2, the percent difference between all manually calculated and MATLAB returned values was determined. For Test Image #1 and #2, Table C5 contains the percent difference between manually calculated percent porosity and MATLAB returned percent porosity. Table C6 contains the percent difference between manually calculated and MATLAB returned total ROI area. Lastly, the percent difference between manually calculated and

MATLAB returned values for total ROI void area are shown in Table C7. The percent difference between all manually calculated and MATLAB returned values were equal to approximately zero. Therefore, the code used to execute the MATLAB defect detection program is correct in the sense that it returns accurate percent porosity, total ROI area and total ROI void area measurements. Moreover, because all manually calculated and MATLAB returned measurements are approximately equal to one another it can be deduced that the MATLAB program detects the ROI of each fin properly. Finally, the program's ability to sequence through and override variables on an image-by-image basis is verified by the return of two unique fin porosity analyses.

Table C5: Percent difference between MATLAB returned and manual percent porosity calculations for Test Image #1 and #2.

-	Percent Porosity: MATLAB Returned	Percent Porosity: Manual Calculation	Percent Difference
Test Image #1	1.7544 %	1.7544 %	≈ 0 %
Test Image #2	0.72098 %	0. 72098 %	≈ 0 %

Table C6: Percent difference between MATLAB returned and manual ROI total area calculations for Test Image #1 and #2.

-	ROI Total Area: MATLAB Returned	ROI Total Area: Manual Calculation	Percent Difference
Test Image #1	34953065.5776 um^2	34953065.6 um^2	≈ 0 %
Test Image #2	34020983.8289 um^2	34020983.8 um^2	≈ 0 %

Table C7: Percent difference between MATLAB returned and manual void area calculations for Test Image #1 and #2.

-	ROI Total Void Area: MATLAB Returned	ROI Total Void Area: Manual Calculation	Percent Difference
Test Image #1	613211.6768 um^2	613211.7 um^2	≈ 0 %
Test Image #2	245284.6707 um^2	245284.7 um^2	≈ 0 %

Academic Vita of Matthew E. Dolack

Matthew.Dolack@gmail.com

EDUCATION

Bachelor of Science in Mechanical Engineering — Graduation: May 2017
Minor in Engineering Mechanics
The Pennsylvania State University, University Park, PA
Schreyer Honors College

Thesis Title: Porosity Detection and Analysis using MATLAB of Thin-Walled Structures Fabricated using Powder Bed Fusion at Different Angles with Respect to the Recoater Blade
Thesis Supervisor: Dr. Timothy W. Simpson

ENGINEERING EXPERIENCE

Researcher & Development Intern — May 2016 – May 2017
Applied Research Laboratory – CIMP-3D — State College, Pa

- Conducted proprietary additive manufacturing research for the US Navy and outside contractors.
- Prepared and held accountable for weekly project summaries to division head.
- Developed software for quantifying internal defects in additively manufactured thin-walled structures.
- Documented the relationship powder bed fusion process parameters have on porosity of thin-walled structures.
- Developed image searching algorithms capable of determining the boundary of a weld melt pool.
    - Program aids in the inline analysis of melt pool dimensions during directed energy deposition processes.

Writing Assistant/Intern — Aug. 2016 – Dec. 2016
Mechanical Engineering Design Methodology — State College, Pa

- Assisted in developing templates for proposals, progress reports, memos and presentations.
- Critiqued and graded course proposals, progress reports, memos and presentations.
- Delivered presentations detailing proper presentation techniques.

TECHNICAL PUBLICATIONS

McDonnell, B., Guzman, X. J., Dolack, M., Simpson, T. W. and Cimbala, J. M., 2016, "3D Printing in the Wild: A Preliminary Investigation of Air Quality in College Maker Spaces", *2016 Annual International Solid Freeform Fabrication Symposium*, August 8-10, University of Texas at Austin, Austin, TX, pp. 2456-2469.