Rose-Hulman Institute of Technology
Rose-Hulman Scholar

Graduate Theses - Mechanical Engineering

Mechanical Engineering

Spring 3-2019

Automated Microstructure Analysis of Large Regions

Cheng Xu

Follow this and additional works at: https://scholar.rose-hulman.edu/dept_mechanical_engineering

Automated Microstructure Analysis of Large Regions

A Thesis

Submitted to the Faculty

of

Rose-Hulman Institute of Technology

by

Cheng Xu

In Partial Fulfillment of the Requirements for the Degree

of

Master of Science in Mechanical Engineering

March 2019

© 2019 Cheng Xu

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Final Examination Report

Cheng Xu	Mechanical Engineering
Name	Graduate Major

Thesis Title Automated Microstructure Analysis of Large Regions

DATE OF EXAM: April 18, 2019

EXAMINATION COMMITTEE:

	Thesis Advisory Committee	Department
Thesis Advisor:	Patrick Cantwell	ME
	Ryder Winck	ME
	Mike McInerney	PHOE

PASSED X FAILED ____

ABSTRACT

Cheng Xu
M.S.M.E.
Rose-Hulman Institute of Technology
May 2019
Automated Microstructure Analysis of Large Regions
Thesis Advisor: Dr. Patrick Cantwell

The mechanical and physical properties of materials are often related to grain size. To accurately measure grain size over a large region, it is imperative to acquire high-resolution image data for each of the grains and detect all of them. Although there is an existing technique, Electron Backscatter Diffraction (EBSD), that can achieve the objective, it is an expensive technique which requires a Scanning Electron Microscope (SEM). Therefore, it is beneficial to develop a less expensive technique that can use traditional and cheap equipment, such as an optical microscope. This thesis is dedicated to making an optical microscope automatically scan over a large region of a specimen and offer microstructural quantification including grain boundary identification, grain size number and grain size distribution. In addition, the maps of grain size and grain aspect ratio will be created to give users a better visualization. The feature of multiple zoom levels for the microstructure is also added for users to navigate through the large regions easily. Finally, software has been developed to help users identify "rare events" in microstructure, such as abnormal grains, and flag those rare events in the map for visual inspection.

Keywords:

automatic, less expensive, optical microscope, microstructural quantification

DEDICATION

This thesis is dedicated to Dr. Cantwell, for without his early suggestions, inspiration, and coaching none of this would have happened.

ACKNOWLEDGMENTS

First, I would like to express my sincere gratitude to my advisor Dr. Cantwell for the continuous support of my thesis, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I can not have imagined having a better advisor and mentor for my thesis.

Besides my advisor, I would like to thank the rest of my thesis committee: Dr. Winck and Dr. Mclnerney, for their insightful comments and encouragement, but also for asking hard questions, which incentivized me to widen my research from various perspectives.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

EBSD	Electron Backscatter Diffraction
SEM	Scanning Electron Microscope

LIST OF SYMBOLS

English symbols

$\bar{l}$	The mean intercept length
L_t	The total line length
P	The sum of the total number of intersections
M	The magnification of the micrograph
D_a	The grain diameter

GLOSSARY

Unimodal -	a probability distribution which has a single peak
Bimodal -	A continuous probability distribution with two different modes

1. INTRODUCTION

Nowadays, it is common that researchers use optical microscopes and computers to help analyze grains in materials. For instance, they usually do quantification of the grain size and aspect ratio since they are critical factors that affect the mechanical properties, such as the yield strength, the ultimate tensile strength, the uniform elongation and the reduction in area [1]. However, when the microscope specimen is much larger than the optical microscope's field of view, it will be difficult and inaccurate if they want to measure the grain size over that region. The reason is that the grain size will only be representative of the relatively small region measured within the field of view and it might not be representative of the rest of the specimen. That can cause inaccurate grain analysis results such as the mean grain size diameter and grain size distribution over the entire region of interest since the grains can vary greatly over a large region. For example, Figure 1 shows images from different locations on the same specimen but the grains are very different in term of aspect ratio. Therefore, obtaining a view of the entire region of interest, especially a large region, is crucial for grain analysis.

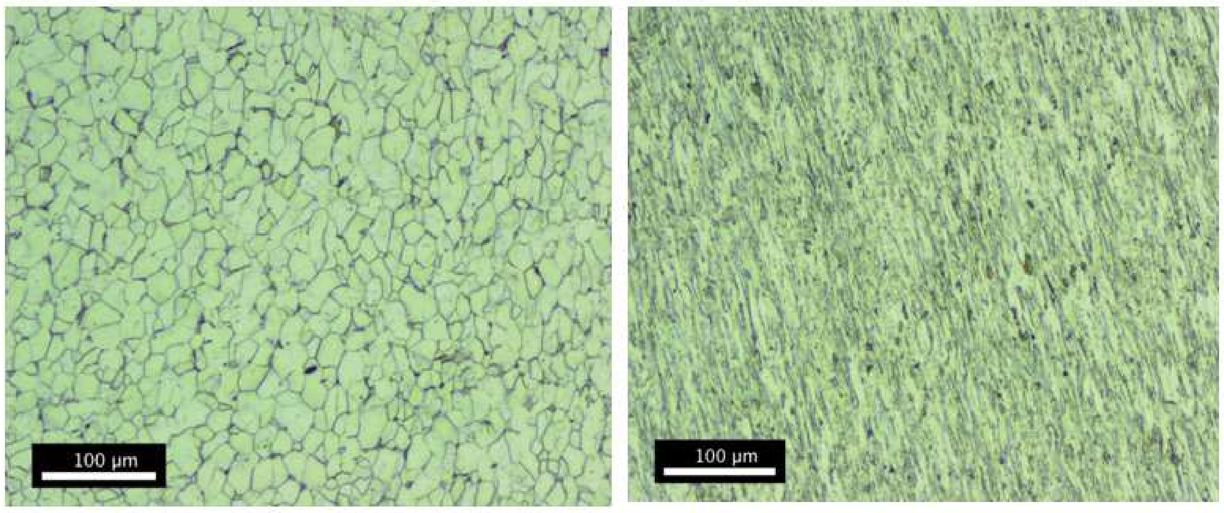


Figure 1: The left figure shows the equiaxed grains at the shaft part of the steel bolt while the right figure shows the elongated grains at the head part of the steel bolt.

It is possible to obtain a view of a large region of interest using an existing technique named Electron Back Scatter Diffraction, EBSD. However, it is time-consuming and expensive to use it. Typically, it can take approximately 10 hours and $2000 to finish collecting all the data on a sample [2]. In addition, EBSD can only be used on a Scanning Electron Microscope, SEM, which is expensive to purchase. Therefore, a materials lab is more willing to use an optical microscope, which is much cheaper than the SEM. However, to acquire a view of a large region with an optical microscope, it is either too expensive to use a commercially available motorized stage or it is inefficient and inaccurate to move an unmotorized stage by hand to collect images over a large region. Hence, it becomes necessary to build a low-cost motorized stage to collect all the images over a large region. A similar problem also exists in the grain analysis software. Although there is an existing software, MagniSci [3], that can help to analyze grains, but it requires some manual work, such as manually closing the partial grain boundaries, to complete the analysis. Besides, all the images collected by the motorized stage need to be stitched together to have a view of the entire region of interest. However, existing software does not analyze large stitched images well, because a large stitched image will typically have low resolution due to compression. Thus, it is necessary to make an algorithm that can analyze a large stitched image and eliminate any manual work.

Therefore, the primary goal for this project is to lower the cost of imaging, quantify the grain size of a large region of interest of a specimen, and automate the process of analyzing its microstructure, such as grain size distribution and grain aspect ratio. In addition, in order to give users a better visualization, the maps of grain size and grain aspect ratio will be created. We will also create maps with multiple zoom levels, like Google Maps, of the microstructure so that the user can navigate through the large regions easily. Finally, a software was developed to help researchers identify “rare events” in microstructure (e.g., abnormal grains) and flag those rare events in the map for visual inspection. The abnormal grains are very useful to the researchers since they are an indirect sign of a grain boundary transition [4] and they also play a role in bulk properties [5]. For example, the yield strength of a metal depends on the distribution and size of grains.

The design details of a low-cost motorized stage replacing the original stage installed on the ZEISS Axio VERT.A1 microscope will be described in this thesis. It replaces the process of moving the stage by hand, which is time-consuming and inaccurate. MATLAB is used for the software design for this project since it is a common engineering programming language, and it helps to integrate the controlling software into the analytical tool. Specifically, MATLAB receives information from a camera installed on the microscope for image acquisition and control purposes. In addition, software routines that perform microstructure analysis are also developed in MATLAB. The entire process is verified by performing grain analysis on both metal and ceramic materials.

2. LITERATURE REVIEW

2.1 Motorized Stage

Currently, almost all motorized stages are used in scientific applications which necessitates precise positioning of the stage with nanometer-scale accuracy. Therefore, motorized stages of this quality are quite expensive due to the cost of the motors and their complicated designs. For instance, a linear motor stage sold by Thorlabs can achieve 100 nm per movement and ±0.25 µm bidirectional repeatability. However, the cost of this stage is $9,904.20 due to the used of a piezo motor used in the motorized stage.

ZEISS provides a motorized stage with stepper motors. However, there is no way to communicate the motorized stage with MATLAB since it has proprietary control software. Therefore, it only works with Zeiss software, and this software does not contain the features we will develop in this thesis. In addition, the motorized stage from ZEISS can only be installed on a specific model of ZEISS microscope, and it is expensive to purchase both the motorized stage and ZEISS software, which costs more than $15000 in total. Therefore, it is necessary to make our motorized stage less expensive and more versatile so that it can work with a variety of software and manufacturers' microscopes with only minor stage design modifications.

Unlike the samples in the biology and life science lab (usually in nanoscale), materials science samples, such as metals and ceramics, are typically much larger (usually in microscale or larger). As shown in Table 1, the smallest dimension 0.085mm, which is 85 µm, is much larger than 100 nm (the minimum movement of the motorized stage sold by Thorlabs). Therefore, there is no requirement for a motorized stage with nanoscale resolution. However, it is difficult to find a cheap but suitable motorized stage with expected accuracy in the market. As mentioned before, most motorized stages are made for biology and life science labs, and they overperform for most analyses in a materials lab. Hence, there might be a market demand for a cost-effective and simple motorized stage which still can meet the requirements for a materials lab. In addition, making the motorized stage compatible with common software is also crucial because it can avoid installing a new standalone application just for the motorized stage which might cause communication problems with the analysis tool (e.g. the data from the stage might not be useful to the software used for the analysis). Therefore, it is necessary to build a motorized stage that is less expensive but precise enough for a materials lab, and it should be designed in the way that any common program used for analysis can communicate with the stage, such as MATLAB.

Table 1: Image dimensions with different objective lens.

Objective Lens	**Relationship between pixel edge length and real distance in the image**	**Image Dimensions (WxH)**	**Actual viewable size per image**
5x	1.800 µm/pixel	1280 x 960 pixels	2304µm x 1728µm
10x	0.900 µm/pixel	1280 x 960 pixels	1152µm x 864µm
20x	0.445 µm/pixel	1280 x 960 pixels	567µm x 427µm
50x	0.175 µm/pixel	1280 x 960 pixels	224µm x 168µm
100x	0.087 µm/pixel	1280 x 960 pixels	111µm x 84µm

2.2 Image Stitching

Image stitching is the process of combining multiple photographic images with overlapping fields of view to produce a large image. Thus, it can allow the user to view a large region on the specimen.

There are many image stitching algorithms such as scale-invariant feature transform (SIFT) [6] and speeded-up robust features (SURF) [7]. They are both feature detection algorithms which detect and describe distinct features in images. Then they match features in a set of images and use direct alignment methods to search for image translations that minimize the sum of absolute differences between overlapping pixels. As described by name, the SURF algorithm is several times faster than SIFT and more robust than SIFT. However, this algorithm is widely used for a 360° view using regular photographs instead of microscope images. Therefore, alternative image stitching algorithms designed for microscope images have been made available.

Image stitching has been used for the microscopy for a long time and there is a professional software, called Microscopy Image Stitching Tool (MIST), which can help stitch the images from a microscope together MIST requires a rectangular array of images and its start point as its inputs. Although MIST can take any kind of images as input, it will convert them to grayscale images automatically. Therefore, it will only output the stitched image in grayscale.

Unlike standard algorithms, MIST developed a novel algorithm for image stitching. Instead of using features matching, MIST computes and optimizes the image translations by the phase-correlation method [8] using the Discrete Fourier Transform. Specifically, if the Discrete

Fourier Transforms of the image A is expressed as $G_a(u,v)$, then if the image B has overlap area with image A, then the image B can be expressed as $G_b(u',v')$ where

$$G_b(u',v') = G_a(u+\Delta u, v+\Delta v)$$

With the phase-correlation method, we can calculate both Δu and Δv, which are the image translations.

Another novel part of this method is that it estimates the microscope stage repeatability from the computed translations of a gridded image tiles to improve the performance of image stitching. It then optimizes all translation computation using the Hill Climbing algorithm [9] bounded to four times the stage repeatability. This minimizes the maximum uncertainty related to the translation computation for any pair of images [10], [11].

However, there is a clear limit when using MIST. The MATLAB version of MIST only accepts gray images as input and outputs a gray stitched image. However, an RGB stitched image is desired for better visualization. In addition, if the region of interest on the sample is so large that it needs to take thousands of images to cover, it will be quite inefficient to collect thousands of images by manually moving the stage over a large region, and people get bored while doing such kind of work. Hence, the image stitching technique itself is not enough and a motorized stage is necessary for data acquisition. Unfortunately, currently, there are very few programs that can automatically control a motorized stage to get all the images from the region of interest and stitch them together. Therefore, a program which can take a region on the sample from a user and output a stitched image covering that region will be useful to the researchers in the lab.

2.3 Image Stitching

One of the primary goals of this project is to quantify grain size, such as grain size distribution and aspect ratio measurement, without any manual work. It is more efficient to convert an RGB or gray microscope image to a binary image before conducting any analysis. The most proper technique that can achieve the goal is called image segmentation because it can simplify and change the representation of a digital image into multiple sets of pixels. In this project, we represent a microscope image using two segments, black and white, as shown in Figure 2. This process is called *grain boundary identification.*

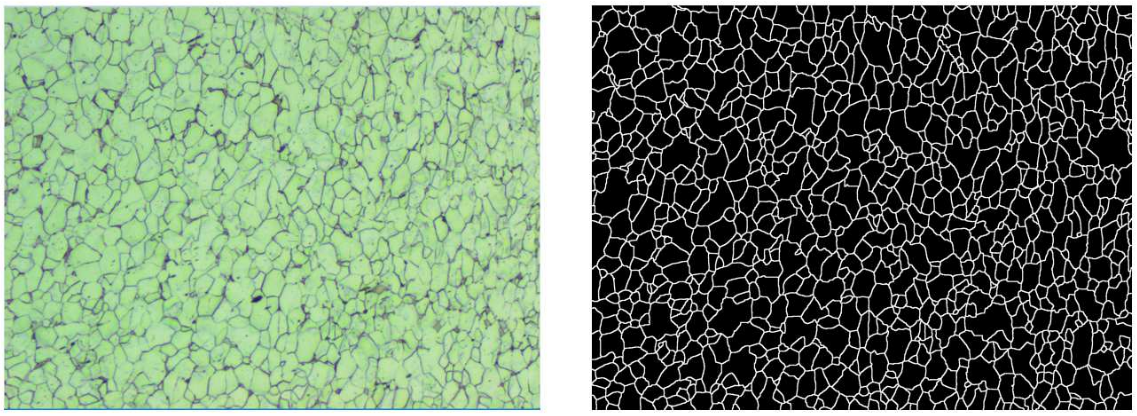


Image before image segmentation
Image after image segmentation

Figure 2: Image segmentation result for a microscope image.

There are many different techniques that can achieve image segmentation, such as Otsu's method [12], K-means clustering [13] and watershed segmentation [14]. We decided to use the watershed segmentation method to separate adjacent grains. The watershed transformation treats a digital image as a topographic map and then finds the lines that run along the tops of ridges. There are many algorithms that can perform watershed segmentation and the Fernand Meyer algorithm [15] is used in MATLAB. The algorithm accepts the gray image as an input and performs based on the idea of watershed by flooding [14]. The watershed transformation first correlates the brightness of each pixel to a height. A water source is placed in each local minimum to flood the entire region from the source. If two different water sources meet each other, a barrier is built at the intersection. This process is demonstrated in Figure 3. After the image segmentation process, the computer can start doing analysis, such as quantifying the grain size, on the black & white image.

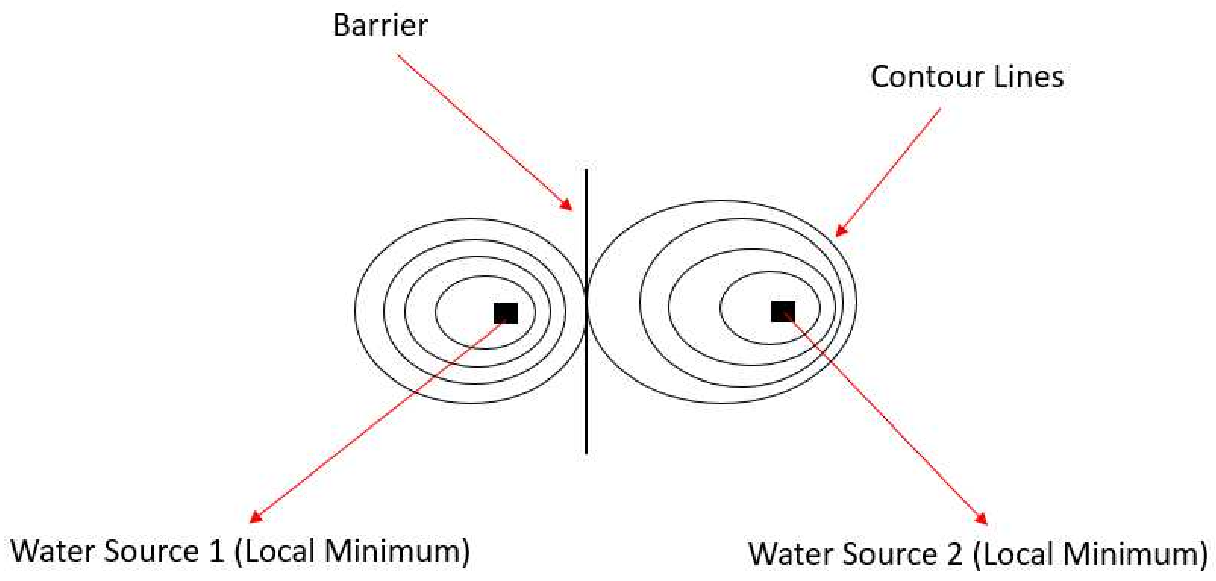

Barrier
Contour Lines
Water Source 1 (Local Minimum)
Water Source 2 (Local Minimum)

Figure 3: Watershed by flooding diagram. Two water sources are placed in this diagram to flood the entire region from the source. a barrier is built at the intersection.

2.4 Grain Size Measurement

The standard of measuring the grains size is given by ASTM E112 [16] and it presents many procedures such as the planimetric (aka Jeffries) or an intercept procedure (e.g., lineal intercept or circular intercept). Figure 4 shows a simplified version of the linear-intercept method from a textbook [17]. First, we draw a bunch of lines within an image of microstructure. For example, seven lines are drawn at the bottom of Figure 4. Then we calculate the total length of all the lines drawn in the image and add all the number of grain-boundary intersections for each line. From here, we can calculate the mean intercept length using the formula:

$$\bar{l} = \frac{L_t}{PM}$$

Where $\bar{l}$ is the mean intercept length, L_t is the total line length, P is the sum of the total number of intersections and M is the magnification of the micrograph. Then the grain size G can be calculated as:

$$G = -6.6457\log\bar{l} - 3.298 \text{ (for } \bar{l} \text{ in mm)}$$

$$G = -6.6353\log\bar{l} - 12.6 \text{ (for } \bar{l} \text{ in in)}$$

For the simplicity of the measurement program, we used lineal-intercept method that is slightly different from the one in ASTM E112 and equivalent circular area diameter method [18]. More details will be explained later.

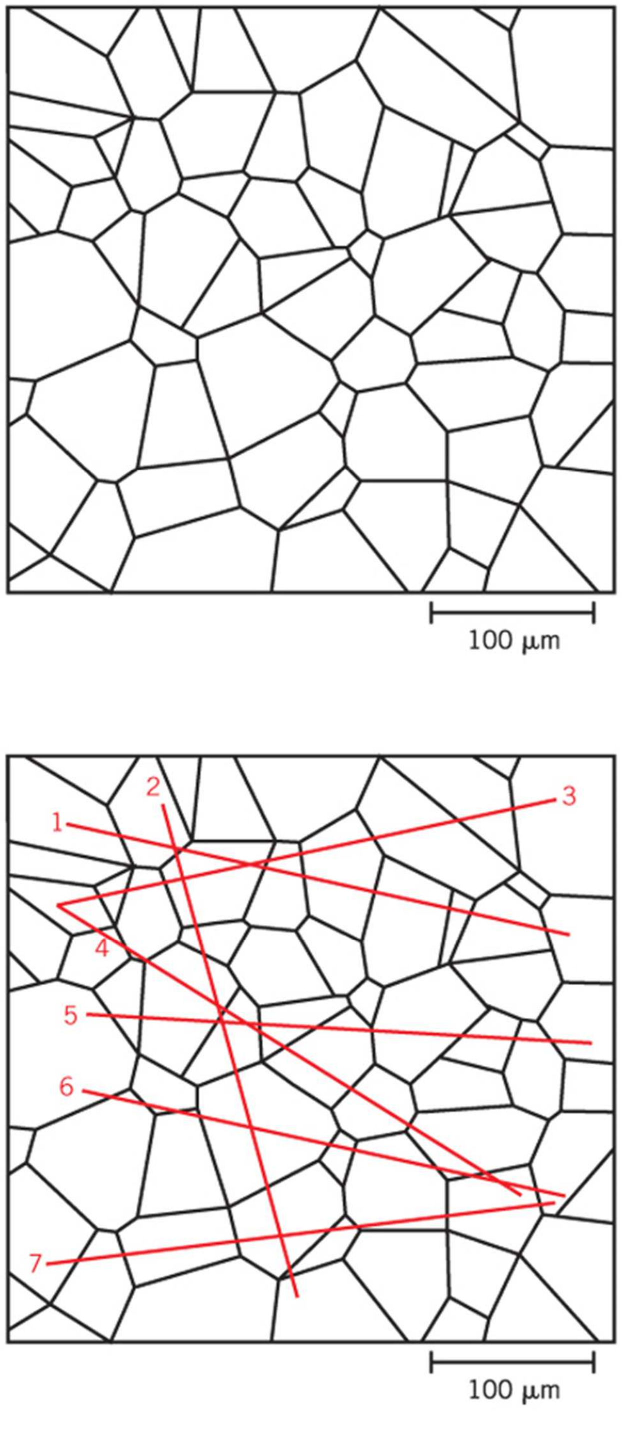


Figure 4: Simplified lineal intercept method [17]

The aspect ratio is another important property for the grains. The aspect ratio is defined as the longest dimension of the grain ("maximum Feret diameter") divided by the Feret diameter measured perpendicular to the maximum Feret diameter [19]. The Feret diameter is the outer dimension of the grain at a given angle as shown in Figure 5. However, there is an easier way for measuring the aspect ratio, which is to first identify a "minimum bounding rectangle" that encompasses the grain as shown in Figure 6, and then divide the long length of the rectangle by the short length [19]. Although they might have different results, they will be very close to each other. From the example given by Figure 5 and Figure 6, the result of "maximum Feret diameter method" can be calculated as 781.13/331.18 = 2.36 while the other one is 767.04/317.08 = 2.42. For the simplicity and efficiency, the "minimum bounding box" method is used in the project.

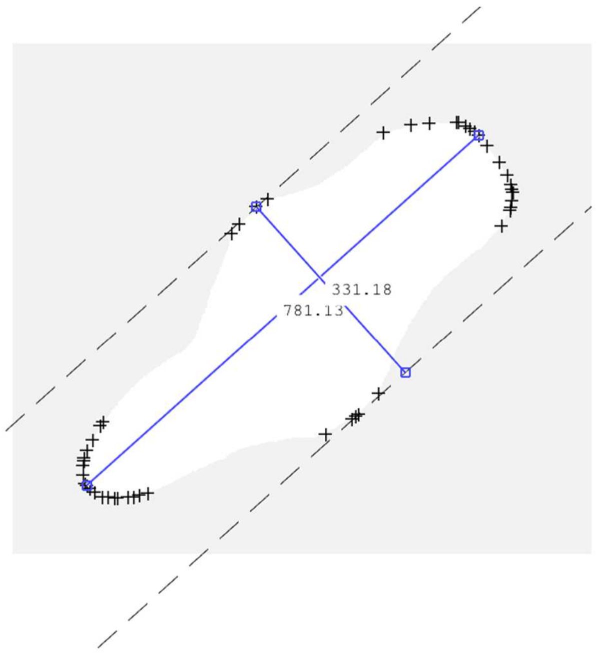


Figure 5: Maximum Feret diameter for a grain. Mark 'x' helps to determine the maximum and minimum Feret diameter.

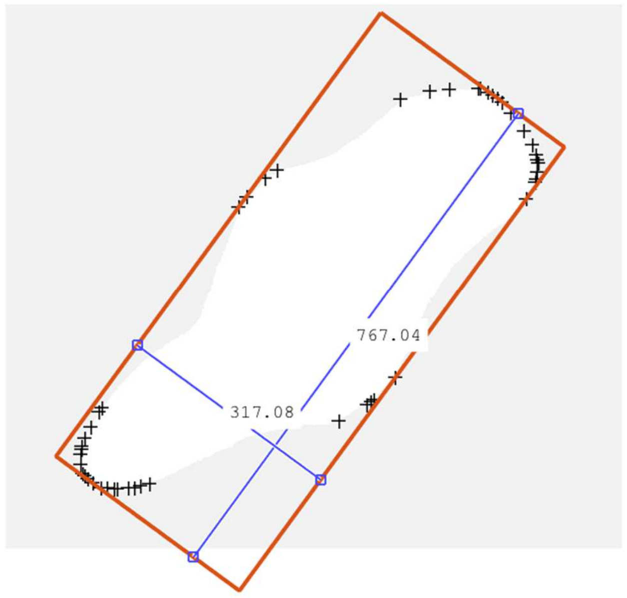


Figure 6: Minimum bounding rectangle for a grain. Mark 'x' helps to determine the minimum bounding box.

2.5 PDF and CDF Plot

The probability density function (PDF), cumulative distribution function (CDF) and grain size map are used for better visualization of grain size statistics. There are two types of PDF. One is called the number-weighted PDF, which specifies the probability of size of a grain falling within a particular range of values based on the number of grains. The other one is called the area-weighted PDF, which calculates the probability of size of a grain falling within a particular range of values based on the areas of grains. Since the grain statistics are discrete, a discrete function will be used for the PDF plot. Therefore, from the PDF plot, it is easy to see the common grain size. In addition, the PDF plot determines whether there is an abnormal grain in the sample since there will be a range with large values appearing if there are any abnormal grains. The CDF plot is a function that specifies the probability of a random variable whose value is less than or equal to the input variable for the cumulative distribution function. For the grain size distribution, the CDF plot does not provide a visualization that is as good as the PDF plot does. The details of grain size map will be explained in a later section.

There are some software packages available to help researchers do grain analysis for materials. Currently, it is still inefficient and inconvenient to use these packages due to the manual work required. For example, figuring out the distribution of grain size in the region of interest with typical software requires time-consuming manual processing by the researcher. These processing steps include identifying the grain boundaries and measuring the grain size. However, automatically identifying the grain boundary and measuring without any input from the researcher is possible. Therefore, we developed software that takes a region of interest on a material sample as input; then it provides the user a view of that region. Once the desired view is obtained, the user can input a command to analyze this region, and our software will automatically output the result data, such as grain size distribution.

3. DESCRIPTION OF THE MOTORIZED STAGE

This section discusses the designs of the motorized stage to replace the original stage installed on the ZEISS optical microscope.

3.1 Original ZEISS Optical Microscope

Figure 7 shows the microscope that we used in this project [20]. The stage (#3 in Figure 7) is unmotorized and can be moved by rotating the knobs (#5 and #6 in Figure 7). Under the stage, there are 5 objective lenses: 5x, 10x, 20x, 50x, and 100x. Rotating the focus knob (#8 in Figure 7) can adjust the height of the objective lens for focus. Since they all have a fixed focal length, the image can be focused by moving the lenses up or down. Our goal is to replace the original stage by a low-cost motorized stage and make the focus knob (#8 in Figure 7) rotate (i.e., raising or lowering the objective lens) automatically for auto-focus. More details will be explained in the stage design section.

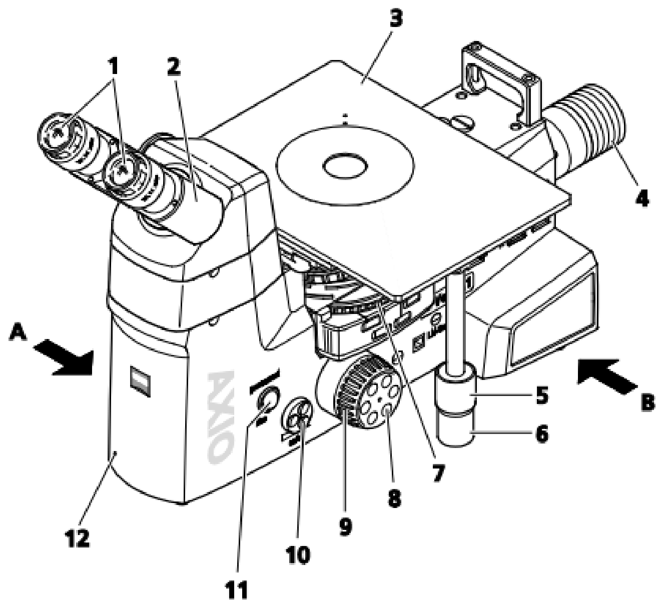


Figure 7: ZEISS Axio VERT.A1 microscope [20]

3.2 Mechanical Design

The ZEISS Axio VERT.A1 microscope with the low-cost motorized stage is shown in Figure 8. We replaced the original stage shown in Figure 7, with a motorized stage and added an auto-focus feature using a stepper motor to turn the focus knobs. Inspired by the design of the ZEISS motorized stage, we designed the low-cost motorized stage with stepper motors and linear bearings.

The dimension of the motorized stage is similar to the original ZEISS stage. The motorized stage contains three plates which bundle together with screws and four linear bearings, which are shown in Figure 9. The engineering drawing of the linear bearing is in Appendix D. The load capacity of the linear bearing is 14 lbs and its straight line travel accuracy is 0.0005" per inch. It can achieve the travel length up to 3 inches. Therefore, the motorized stage is able to scan a specimen up to 3’’ x 3’’, which is larger than both the typical sample size examined in a materials lab and the maximum area that the original stage can scan.

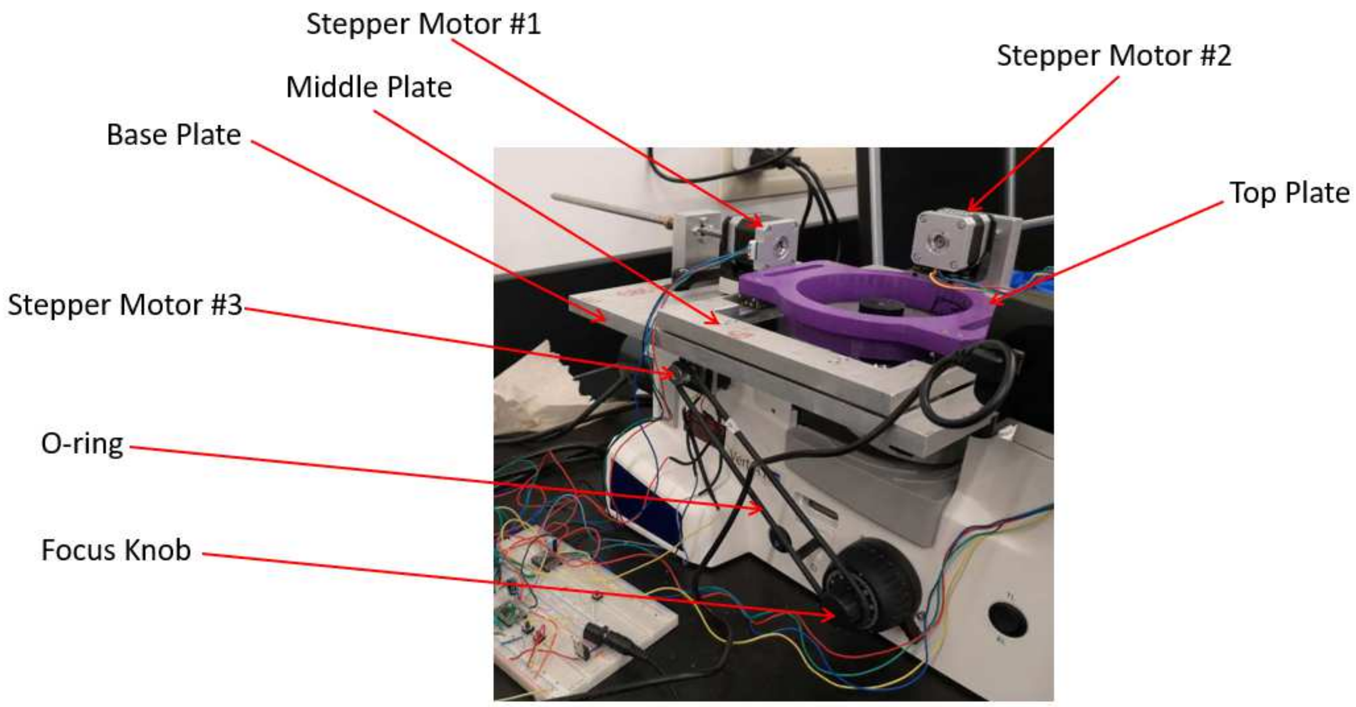


Figure 8: The ZEISS Axio VERT.A1 microscope with the low-cost motorized stage

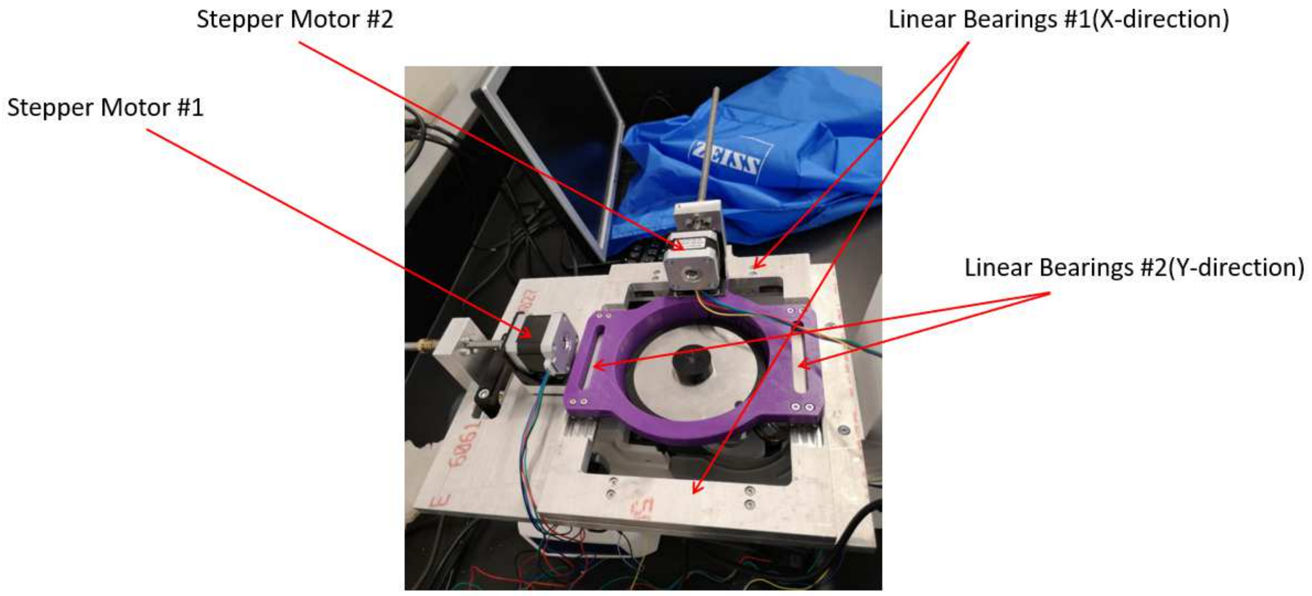

Stepper Motor #2
Linear Bearings #1(X-direction)
Stepper Motor #1
Linear Bearings #2(Y-direction)

Figure 9: More details for motorized stage

3.2.1 Stepper Motor

The choice of the type of the stepper motor for the X-Y direction (i.e., stepper motor #1 and #2) is made based on price, dimensions, the linear bearing's load capacity and the weight of the motorized stage. The dimensions of the stepper motor cannot be too large because it might block the top plate if it is too large (as shown in Figure 8 and Figure 9).Since the linear bearing's load capacity is 14lb, it means that it can carry a very large and heavy motor. Another factor we need to consider is that the output from the stepper motor must be larger enough so that it is able to move the stage. The stepper motor we used for this project is a NEMA-17 stepper motor. Its weight is only 10 oz, and the size of it is only 42.3 mm × 42.3mm × 38 mm without the shaft and lead screw. It only costs $16.95. It can achieve a torque output of 3.7 kg-cm. Since the weight of the motorized stage is only about 1kg, it should move the stage without any problems. The lead screw is added on the shaft of the stepper motor to translate the rotation to linear movement. Therefore, the NEMA-17 stepper motor is able to move the stage linearly via the lead screw.

Although the dimensions of stepper motor #3 for the Z direction is limited by the size of the base plate, it does not need to be a large stepper motor since it only needs to carry the loads from the focus knob and O-ring, which are much smaller than the load of the motorized stage. Therefore, we decided to use a NEMA-14 stepper motor, which is smaller and cheaper than the NEMA-17 stepper motor, as the stepper motor #3. The size of it is only 35 mm × 35mm × 28 mm without the shaft, and lead screw and the weight of it is 5 oz. The cost is only $8.68.

3.2.2 Base Plate

The base plate is at the bottom of the motorized stage and it connects to the ZEISS microscope directly with three bolts. The two notches on this plate are for the linear bearings #1, which allow for motion in the X-direction. The engineering drawing of the base plate is shown in Appendix A.

3.2.3 Middle Plate

The middle plate is at the middle of the motorized stage and it connects to the linear bearings #1 placed on the base plate. The two notches on this plate are also for linear bearings #2, which allow for motion in the Y-direction. The bulge at the center of the edge is for the nut bracket whose purpose is to fix the stepper motor #1. The stepper motor #1 moves the middle plate of the stage in the X-direction by turning the lead screw since it is installed on the middle plate which connects to the linear bearings #1. The engineering drawing of the middle plate is shown in Appendix B.

3.2.4 Top Plate

The top plate is at the top of the motorized stage and it connects to the linear bearings #2 placed on the middle plate. It is the stage that will hold the original Zeiss stage insert, which holds the specimen.   The stage insert has an aperture (hole) for the microscope specimen. In addition, it also holds another nut bracket as shown in Figure 9. The stepper motor #2 is fixed on the top plate. Therefore, it moves the top plate of the stage in the Y-direction since it is affixed to the top plate which attaches on the linear bearings #2. The engineering drawing of the top plate is shown in Appendix C.

3.2.5 Mechanics for Auto-focus

The stepper motor #3 is attached on the bottom side of the base plate and it is used for rotating the focus knob so that it can achieve the auto-focus goal. An L-bracket is used so that the stepper motor #3 can connect with the base plate. The connection scheme is shown in Figure 10. As shown in Figure 11, a pulley is installed on the shaft so that an O-ring can be placed. The

O-ring connects the focus knob and stepper motor #3 together so that when the shaft rotates, the focus knob will rotate as well to raise or lower the objective lenses, which focuses the image.

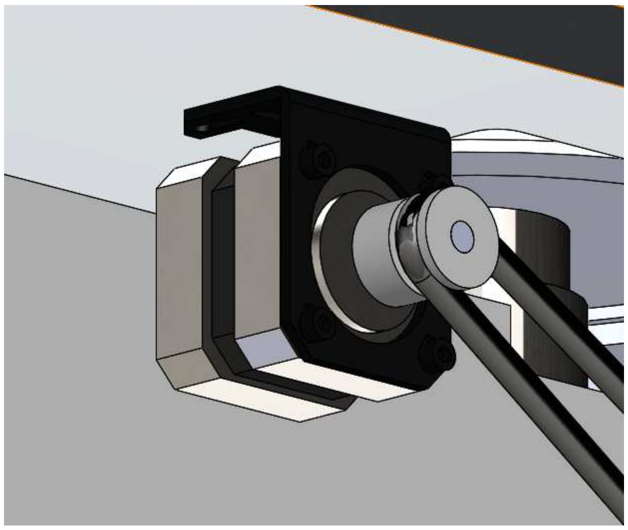


Figure 10: Stepper motor #3 connection scheme. A black mounting bracket holds the motor to the bottom of the base plate. A rubber O-ring connects a pulley on the motor shaft to the focus knob on the microscope.

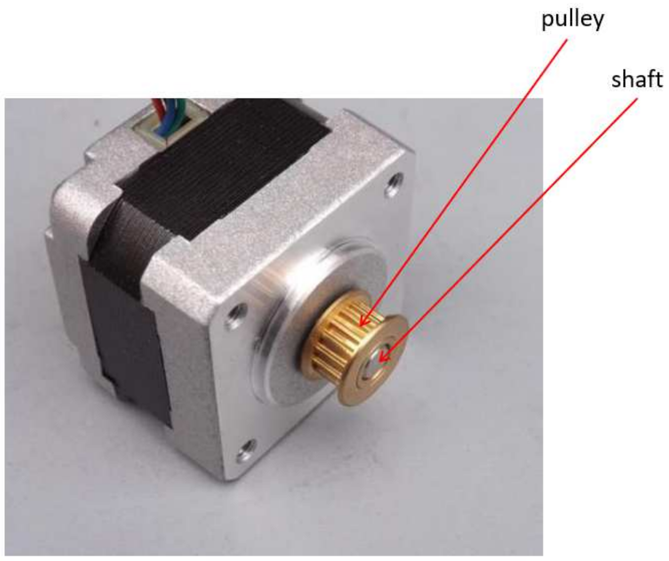


Figure 11: Stepper Motor #3 with pulley installed

3.3 Field of View vs. Region of Interest

As shown in Figure 12, the right image shows the cross-sectional area of a bolt, while the left image shows the field of view of the microscope. It is obvious that the field of view is much smaller than the entire region of the cross-sectional area, and it contains so little information that it is impossible to know the distribution of the grain size over the entire cross-sectional area. In addition, as mentioned in the Introduction section, gains can vary over a large region. For better visualization of how grains vary over a large region, a stitched image is necessary.

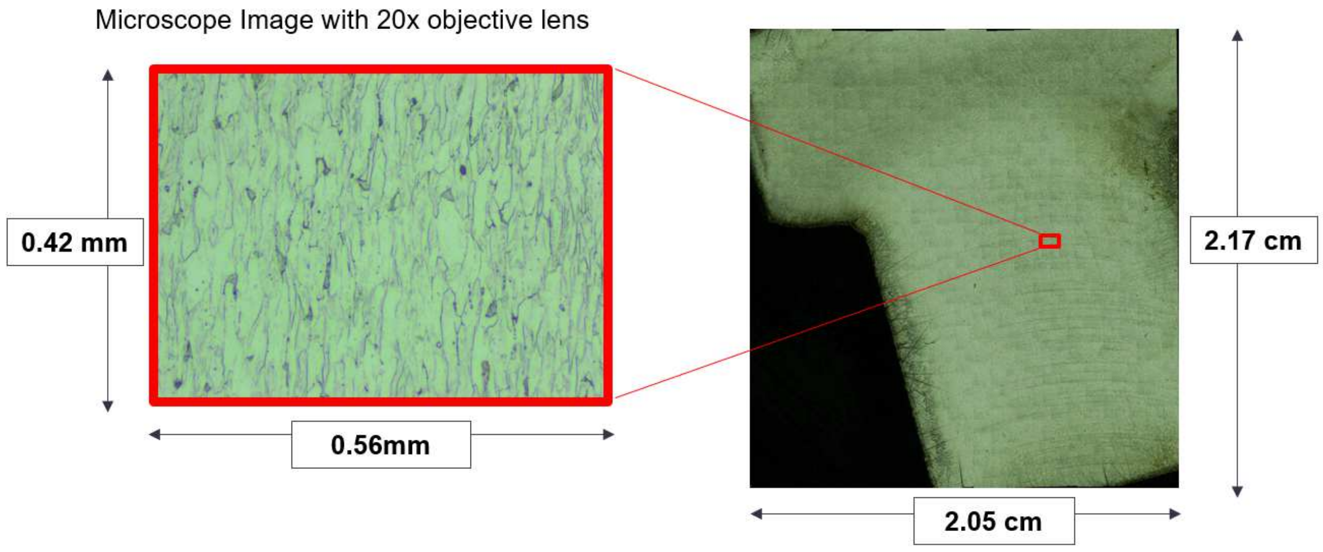


Figure 12: Field of View vs. Region of Interest

3.4 Control Design

3.4.1 **Objective**

The design goal is to acquire overlapping images in a serpentine pattern and to ensure each image is focused. Thus, two stepper motors need to be controlled so that they can make the stage move in the X- and Y-directions to get to each image location. Additionally, the third stepper motor needs to be controlled so that it can rotate the focus knob to move the objective lenses in the Z-direction to focus the image.

3.4.2 Image dimension measurement

To measure the image dimensions the calibration scale for the Amscope camera was used to get the actual pixel size, whose shape is square, in the image. Then, the actual lateral dimensions were calculated based on how many pixels were in a row of the image since the lengths of two sides of a pixel are equal. Figure 13 shows an image of the calibration scale from 50x objective lens. It illustrates how dimensions in the image features relate to the pixels in the image. From Figure 13, the lateral dimension of the pixel can be calculated as

$$\text{lateral dimension of the pixel} = \frac{L}{N} = \frac{50\ \mu\text{m}}{286\ \text{pixels}} = 0.175\ \mu\text{m/pixel}$$

where *L* is the distance between the major line in the scale and *N* is the number of pixels between the major line in the scale.

The image dimensions in all other four objective lenses can be calculated by acquiring the image like the one in Figure 13. The results for all the objective lenses are shown in Table 1.

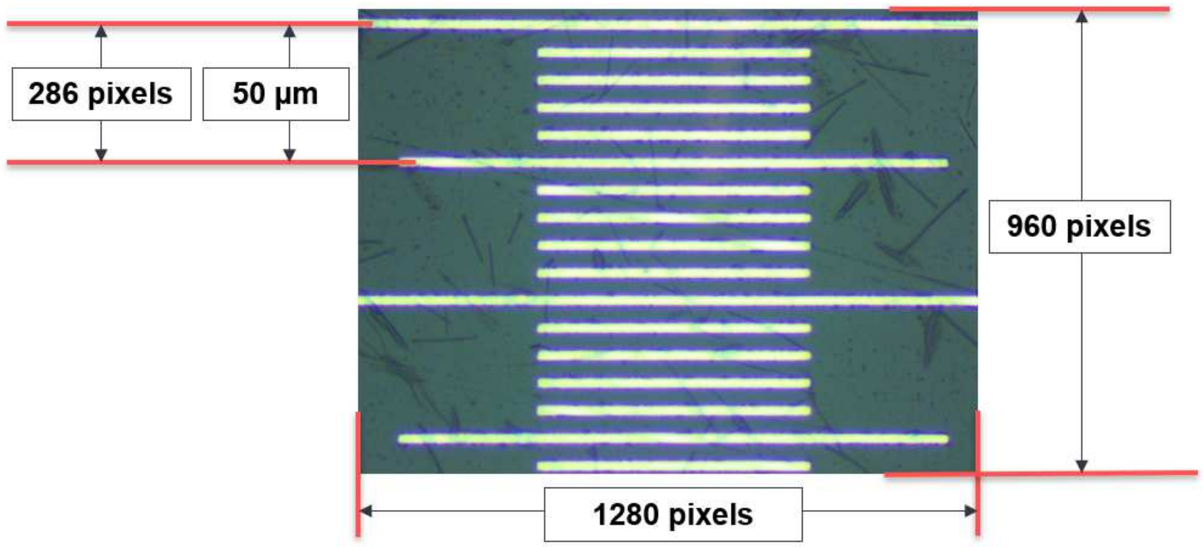


Figure 13: Microscope image of the calibration scale from 50x objective lens used to determine the lateral dimensions represented by a pixel in the image, i.e., the scale of the image. Major lines in the scale are 50 micrometers apart, and minor lines are 10 micrometers apart.

3.4.3 X-Y Dimension Control

As mentioned before, two NEMA-17 stepper motors were used for X-Y dimension moving (one moves the middle plate while the other one moves the top plate) since they are cheap and precise enough to achieve the goal of the control part. The stepper motor is bipolar, and the stepper motor shaft makes one full revolution in 200 steps, which gives a rotation angle of 1.8 degrees per step. To turn the rotation movement into linear motion, we combined the stepper motor with Tr 8 x 4 lead screw, which has a diameter of 8 mm and a pitch of 4 mm, and "Tr" mean the trapezoidal thread. Therefore, one full rotation of the stepper motor moves the stage by 4 mm.

DRV8825 Stepper Motor Driver is used for controlling the stepper motors. This driver allows the stepper motor to move in sub-step increments. For example, the smallest possible increment with this driver is 1/32 step, which is an angle of 0.05625 degrees. Based on the formula for calculating the length of distance traveled,

$$d\text{istance traveled} = \frac{\text{thread pitch}}{\text{total sub steps per revolution}}$$

which makes the NEMA-17 stepper motor minimum movement:

$$\text{distance traveled per 1/32 step} = \frac{4000\ \mu\text{m}}{200\ \text{steps*32 substeps per step}} = 0.65\ \mu\text{m}$$

To make sure that the motorized stage can take overlapping images, the requirement for the minimum movement of the stage in both X and Y directions must be less than the image dimension with the highest objective lens (100x) since the higher the objective lens, the smaller the image dimensions. Therefore, the DRV8825 Stepper Motor Driver is precise enough since it can take an image with the 100x objective lens, then move the stage in X or Y direction for 0.65 microns and take a second image. These two images will have a 99.23% overlap region.

Although adding the feedback control loop for the X-Y dimension seems necessary due to the error caused by the stepper motors (e.g., a stepper motor cannot move the constant distance for each step), it is not an ideal feature to add since it can take up a lot of time. Without a physical sensor installed on a stepper motor, the feedback signal must be generated by calculating the size of the overlapping region of the images at the current location and the expected location. However, this procedure is time-intensive if the X-Y dimension feedback signal needs to be generated frequently. Due to the time issues, the X-Y dimension feedback feature is not used. Although the errors caused by stepper motors are ignored in this thesis, these errors actually can be compensated by the image stitching process except the errors at edges.

Figure 14 shows a stitched image with black "stair step" area, which is caused by the fact that stepper motors are not moving straight, perfectly, in the desired directions.

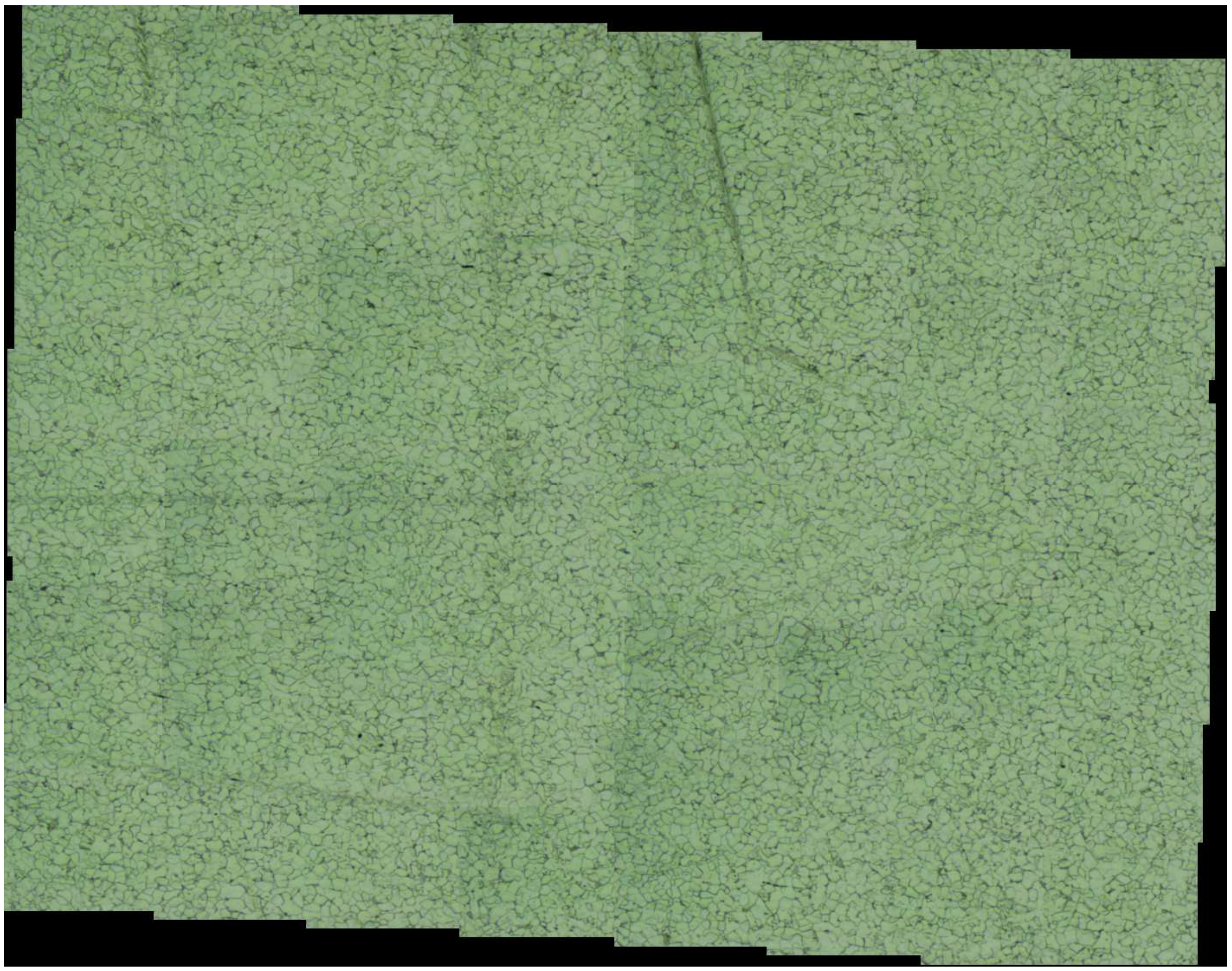


Figure 14: A stitched image with black "stair step" area, which is caused by the errors of the stepper motor at the edges of the entire region.

3.4.4 Z-Dimension Control (Auto-Focus)

Auto-focus is needed because the surface of the sample is not perfectly flat and parallel to the focus plane. This means that the image can become out of focus as the stage is moved in the X- and Y-directions. Therefore, it is necessary to control the focus knob to adjust the height (Z-direction) of the objective lens for auto-focus.

We used the camera from Amscope, model MU-5500. It is the eye for the motorized stage and MATLAB has drivers to control this camera. It uses a CMOS sensor and transmits the image data via USB3.0. It is also the key component for Z-dimension feedback control. The highest resolution for this camera is 2592 x 1944. However, for efficiency, 1280 x 960 resolution was used for image acquisition.

To determine whether an image acquired is out of focus or not, the gradient is a good tool to use since it can tell us how much difference there is between two neighbor pixels. For example, if we are calculating the gradient in X-direction, from the standard gradient formula

$$\nabla G = \frac{\Delta f}{\Delta x}$$

We treat Δf as the difference of the pixel value and $\Delta x = 1$ since they are neighbor pixels. If an image is out of focus, it means there are not many differences between two neighbor pixels. On the other hand, if an image is in focus, there will be many differences at least at edges.

Specifically, the program converts an input RGB image to a gray image to calculate the gradient. Then, the gradient of the gray image in both X-and Y-direction can be calculated by MATLAB function gradient(). After the gradient is calculated, a new variable *sharpness* will be calculated based on the gradient. The formula to calculate the sharpness is

$$S = \frac{\sum_i^n \sum_j^m \sqrt{{Gx_{ij}}^2 + {Gy_{ij}}^2}}{n \cdot m}$$

Where G_x is a matrix with all the values of the image gradient in the X-direction, G_y is a matrix with all the values of the image gradient in the Y-direction, n is the number of pixels in each row of the image and m is the number of pixels in each column of the image. Therefore, the variable S is finally used to determine whether an image is out of focus or not.

Figure 15 shows an example that verifies the value of sharpness is reasonable to tell whether the sample is in focus or not.

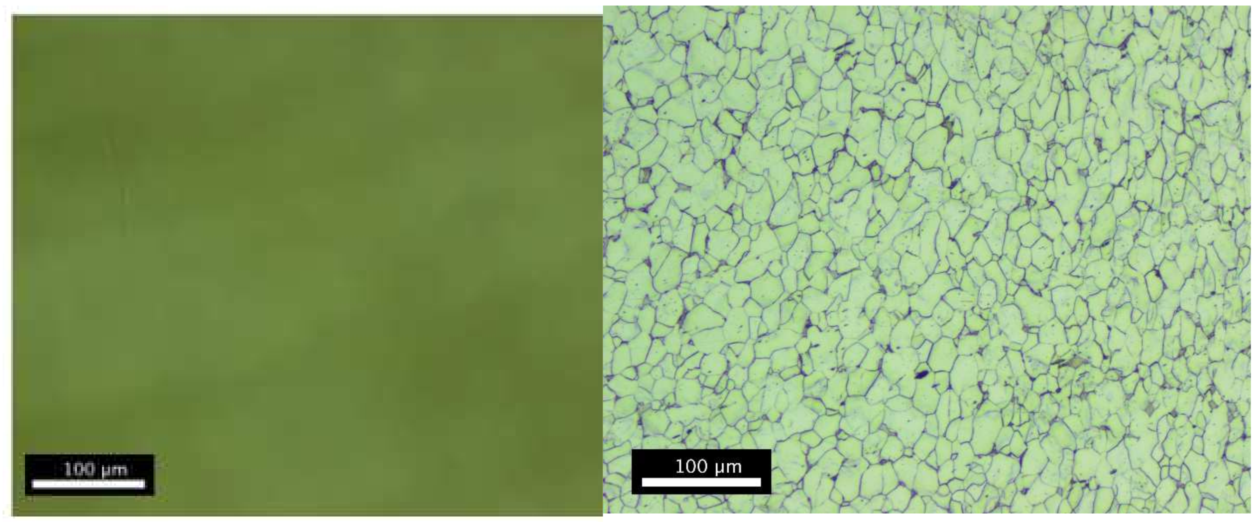


Figure 15: The left side shows the image of the sample out of focus and its sharpness is 1.4. The right side shows the image of the sample on focus and its sharpness is 4. Therefore, they have huge difference in term of sharpness and sharpness can be used to tell whether the sample is on focus or not.

Before Z feedback control can work, a user must manually focus on at least one spot on the sample and the program will use the best sharpness as the ideal sharpness, which will be used to auto-rotate the focus knob. The workflow of Z feedback control is as follows:

1. Receive the image from the start point and set a variable F to 0 if the stage is at the start point.
2. Calculate the brightness of the input image to tell whether the image is from resin or not. A preset threshold is used in this step. If the brightness is above the threshold, the program will go to step 3. Otherwise, the program will set a flag F = 0 and the motorized stage will move to the next location since it is an image from the resin.
3. Calculate the image sharpness first. If F = 0, the program will go to step 4. Otherwise, the program will go to step 5.

4. If the image sharpness is above the preset threshold mentioned before, the program will set F = 1 and go to step 5. Otherwise, the motorized stage will move to the next location since it is an image from the metal sample holder.
5. MATLAB gives the signal to rotate the focus knob 3.3 degrees in an arbitrary direction first. If MATLAB gets a better sharpness, it will go to step 6. Otherwise, it will rotate 6.6 degrees back. If MATLAB gets a better sharpness, it will go to step 6. Otherwise, the motorized stage will move to the next location.
6. MATLAB gives the signal to move the motor according to the control scheme, which will be explained in the next paragraph.
7. The motorized stage will move to the next location if the scheme is satisfied.

Based on the experiments, the control scheme for auto-focus is listed below and works quite well:

- If the current sharpness is larger than 90% of the ideal sharpness, it means this image is in focus. There will be no control signal from the program.
- If the current sharpness is within the range of 75% - 90% of the ideal sharpness, the program will rotate the focus knob 3.3 degrees for all the objective lens.
- If the current sharpness is within the range of 50% - 75% of the ideal sharpness, the program will rotate the focus knob 6.6 degrees for all the objective lens.
- If the current sharpness is within the range of 25% - 50% of the ideal sharpness, the program will rotate the focus knob 22 degrees for all the objective lens.
- If the current sharpness is less than 25% of the ideal sharpness, the program will rotate the focus knob 55 degrees for all the objective lens.

The purpose of step 5 described before is to solve the problem that originates from the fact that sometimes the program cannot determine whether an image that has low sharpness is out of focus or not. Specifically, the potential issues in the process of auto-focus can be that the program might try to focus an image that has already been focused or even manage to make a focused image out of focus. These issues are also the main contributor to making auto-focus the most time-consuming aspect the control process. These issues originate from the fact that the sharpness not only depends on the whether the image is focused or not, but also depends on the patterns of the current image. This means that images of two different regions in the sample could both be perfectly in focus yet have different values of sharpness if they have different microstructural features. Figure 16 shows an example. Thus, the program might need to try many times to figure out what is the value of sharpness for the focused image if the pattern in this image is quite different from the previous images.

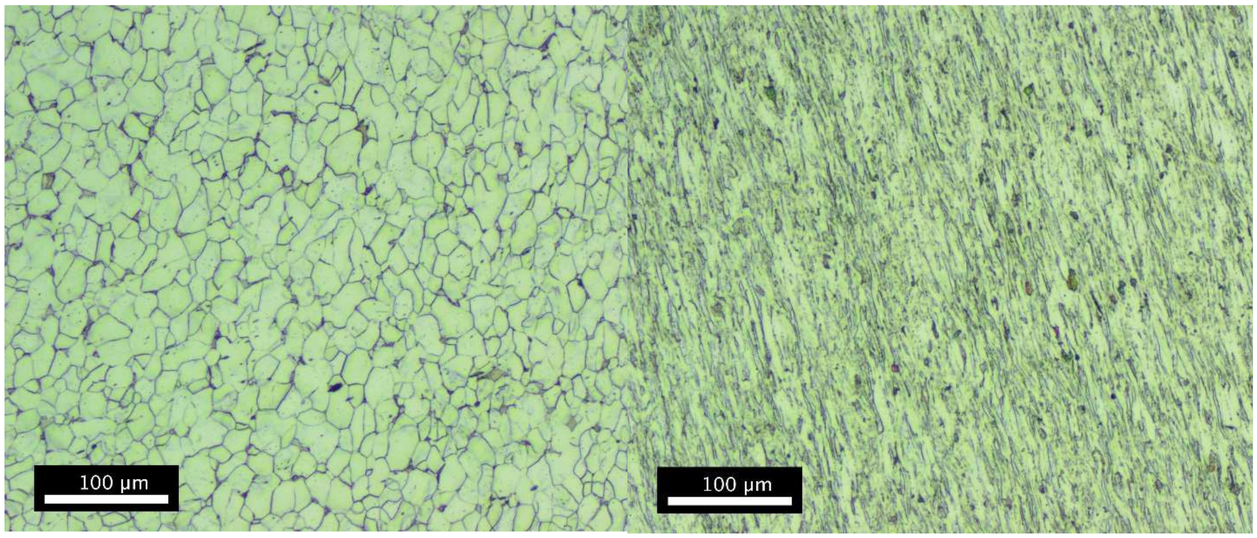


Figure 16: The two in-focus images which have very different sharpness values. The sharpness of the left image is 4 while the sharpness of the right image is 6.5. It means there is 62.5% difference between them.

Therefore, step 5 becomes necessary. Once the program gets such an image, it will first try to make a tiny rotation of the adjustment in both directions to check whether a rotation can help to obtain a better sharpness. Thus, if such an image is already in focus, neither direction will help to have a better sharpness value. Then, the program will know there is no need to focus on this spot and move on.

3.4.5 Image Acquiring Scheme

In order to get a stitched image from a region that a user is interested in on the sample, we designed a "corner stitch" method (shown in Figure 17**Error! Reference source not found.**). First, a user needs to focus on a certain spot on the sample and the program will help the user to calculate the sharpness value S via the formula shown before. Then, a user can input the four corners that will define the region a user is interested in. To input the four corners, a user must move the stage to the desired location and acquire the image four times. A user must ensure that the image taken from the last location must be in focus. There is a technical reason why it is necessary, which will be explained in the next section. The program can also help the user to do the focus work.

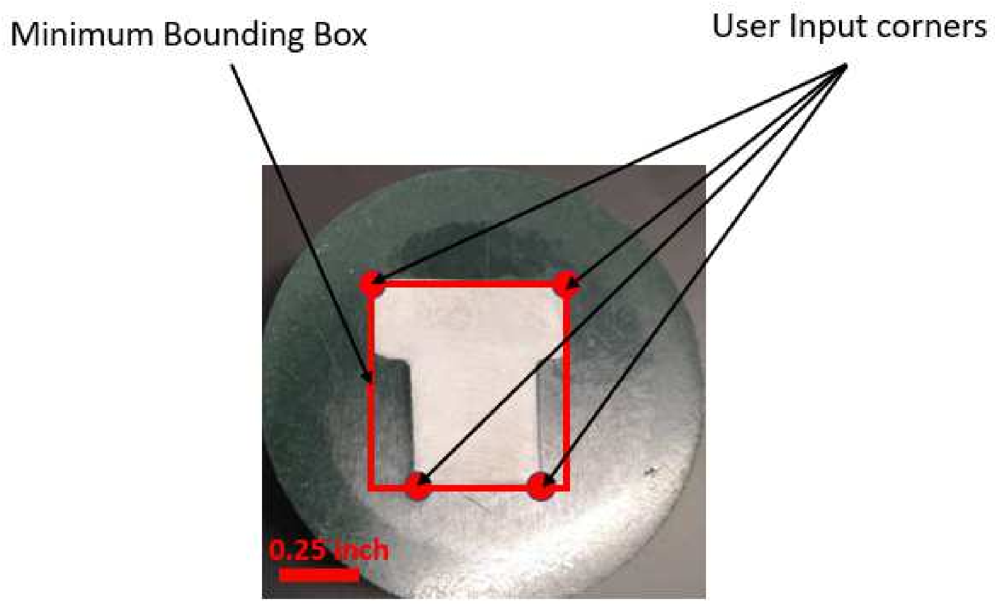


Figure 17: The "T" shape area in the resin is the cross-sectional area of a grade 2 steel bolt. Four corners of the "T" shape region are the inputs to the software and the software will begin to control the motorized stage and camera to scan and acquire the images step by step over the entire minimum bounding box in a serpentine pattern.

After the program obtains an image at all four corners, the program will try to find the minimum bounding box around this region, because all four corners might not be in the shape of a square or rectangle. Then the program will make the stage move to the nearest corner point of this minimum bounding box and set it as the start point. Then the stage will begin to move step by step in serpentine pattern until it finishes scanning the whole minimum bounding box. The program will make sure that each image has an overlapping area by controlling each step size and focusing those regions that are out of focus. At the end, the program will stitch all the images to yield a final result. The corner stitch scheme might fail while a user is trying to stitch a large region. Figure 18 helps to visualize this problem.

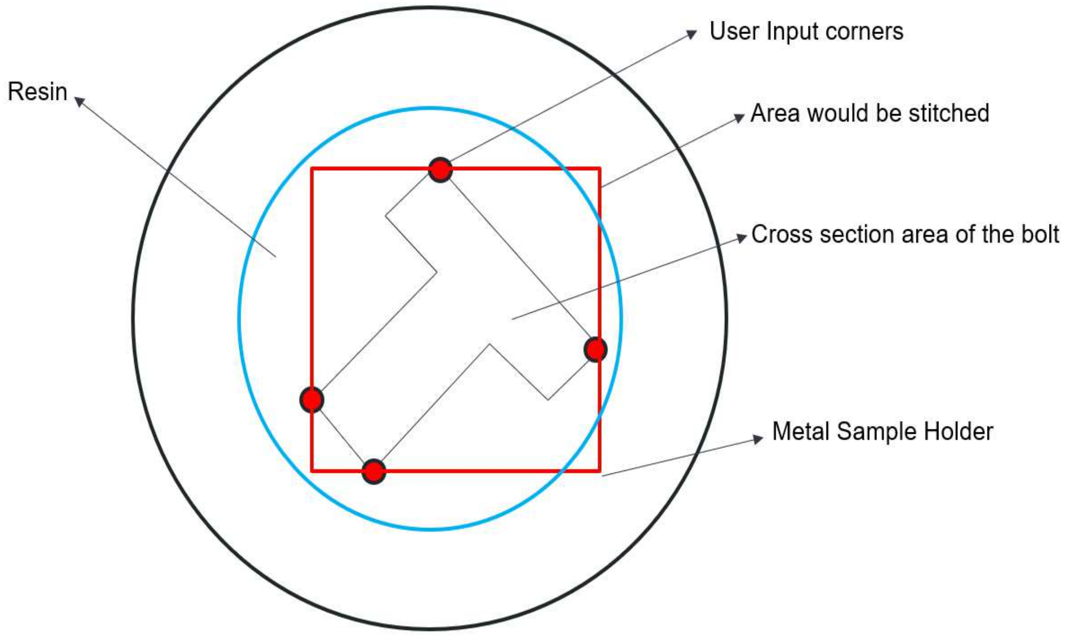


Figure 18: Corner Stitch Problems Visualization

The metal sample holder can cause auto-focus issues since the program cannot tell whether it is the image of the metal sample holder or it is the image of the sample that is out of focus. Since the shape of the entire region of interest might not be square or rectangular, the program will try to find the minimum bounding box around this region. This might cause the problems that part of the minimum bounding box might include the metal sample holder and then cause the auto-focus issues. Specifically, the image of the metal sample holder will be very similar to the left side of Figure 15. It can cause the problem that the focus knob will need to move a lot to focus on the metal sample holder and then move back to refocus on the sample. This will waste plenty of time and can also lead to autofocus failure due to the large movements involved. Hence, the program must ignore the out-of-focus image if it is from the metal sample holder. This explains that the flag F is created in the process of Z feedback control to determine whether an out-of-focus image is from the metal sample holder or not.

A more serious problem can happen if the starting point is within the region of the metal sample holder. Then it will become impossible to finish the stitching process since the program will try to focus on the metal sample holder instead of the sample itself. Eventually, it might make the region of interest totally out of focus.

Therefore, a focused image is necessary at the last input location of the stitched image because the stage will move to the nearest corner point of the minimum bounding box that the program finds from the last input location. If the first image has low sharpness below a pre-set threshold, the program will know that it is an image of a region in the metal sample holder instead of the sample itself. In addition, every time the stage moves to the region of resin, which will give the program a black image, the program will not try to focus the spot which has lower sharpness than the threshold since it is not the region on the sample. This action is valid since it is impossible to have such a big difference in terms of sharpness between the neighboring regions on the sample if the region nearby is already in focus.

3.5 Electrical Design

The overall electric design is shown in Figure 19. The MATLAB sends signals to the Arduino board through the serial port. There is a driver [21], which is a MATLAB support package for Arduino hardware, that allows MATLAB to communicate with Arduino Uno R3 boards.

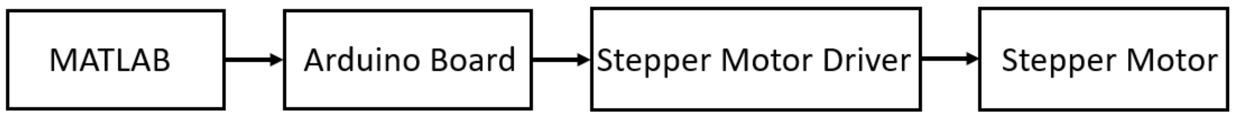


Figure 19: The overall electric design

Two Arduino Uno R3 boards (as shown in Figure 20) are used to control the three stepper motors. The Arduino boards are connected to the step motor driver, which controls the step size and the direction of the stepper motor. Digital Pin5 and Digital Pin6 on the Arduino Uno are connected to the port named “STEP” on the stepper motor driver (shown in Figure 21), which controls NEMA-17 stepper motor. Digital Pin5 of the other Arduino board is connected to the same port on the stepper motor driver which controls NEMA-14 stepper motor. PIN5 and PIN6 are used since they are the pins on the board that have the highest PWM frequency, which is 980HZ. This frequency will make sure that the stepper motor moves smoothly without obvious halt. The wiring diagram for the stepper motor driver is shown in Figure 21, and the microcontroller in the diagram is the Arduino Uno R3 board.

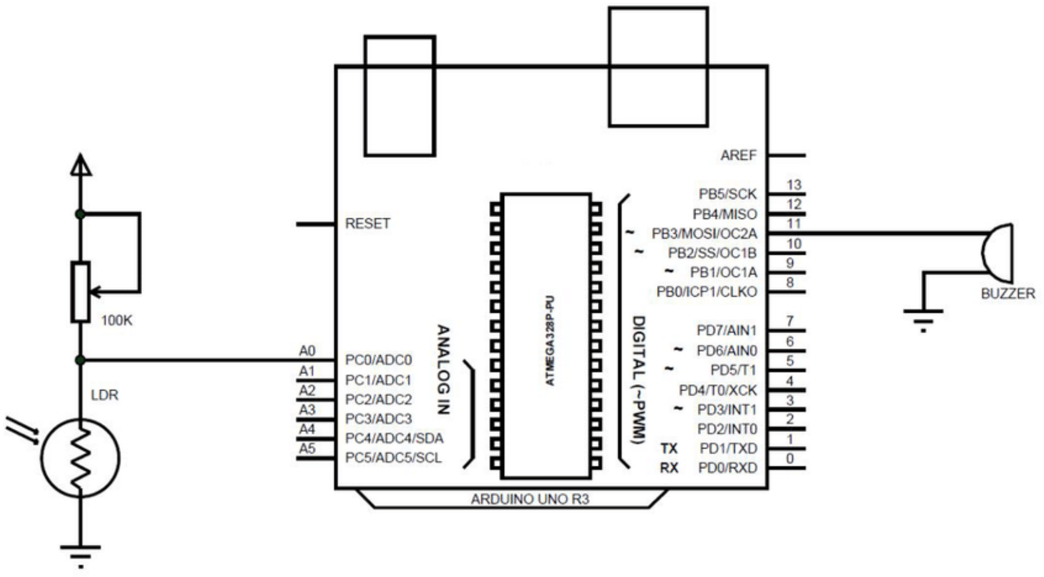


Figure 20: Arduino Uno R3 Diagram

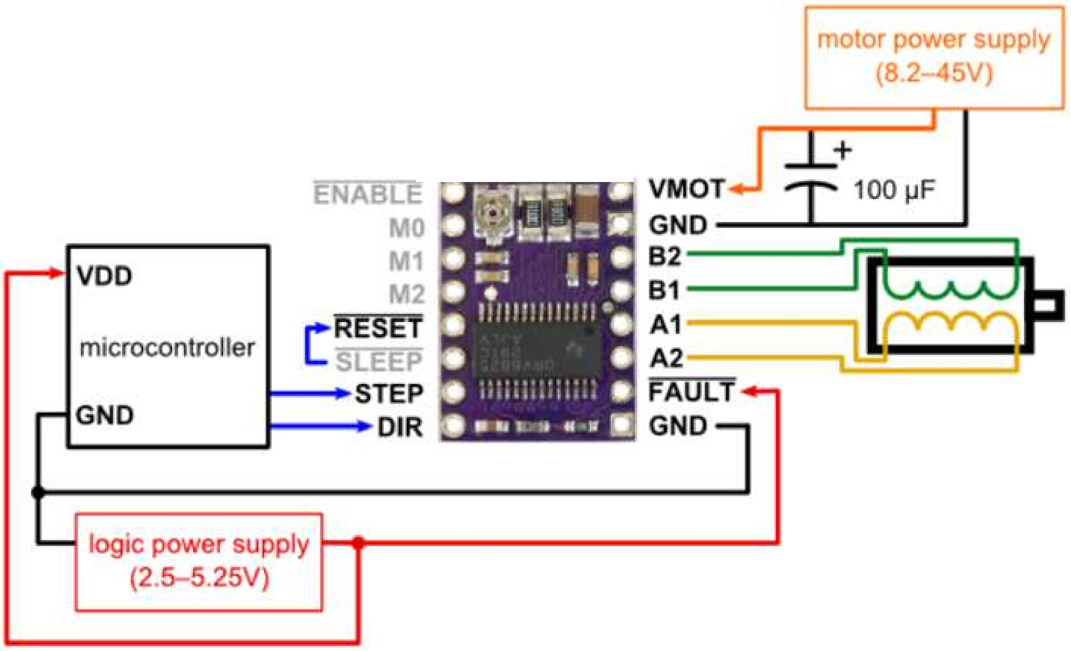


Figure 21: Wiring diagram for DRV8825 stepper motor driver

4. SOFTWARE DESIGN

4.1 MATLAB Program GUI

Figure 22 shows the MATLAB Program GUI. This GUI controls the motorized stage, image acquisition, image stitching, multiple zoom level map features, and image analysis. A user can use this GUI to manually move the motorized stage, automatically perform an image acquisition from the region of interest, stitch any kind of microscope images obtained from other sources, navigate a stitched image using multiple zoom level map features with grain size statistics and conduct analysis on a microscope image. In the following sections, details of each of these subprograms will be explained.

The author wrote the all the software, except basic image stitching program develop by MIST and the program that implements the linear-intercept method.

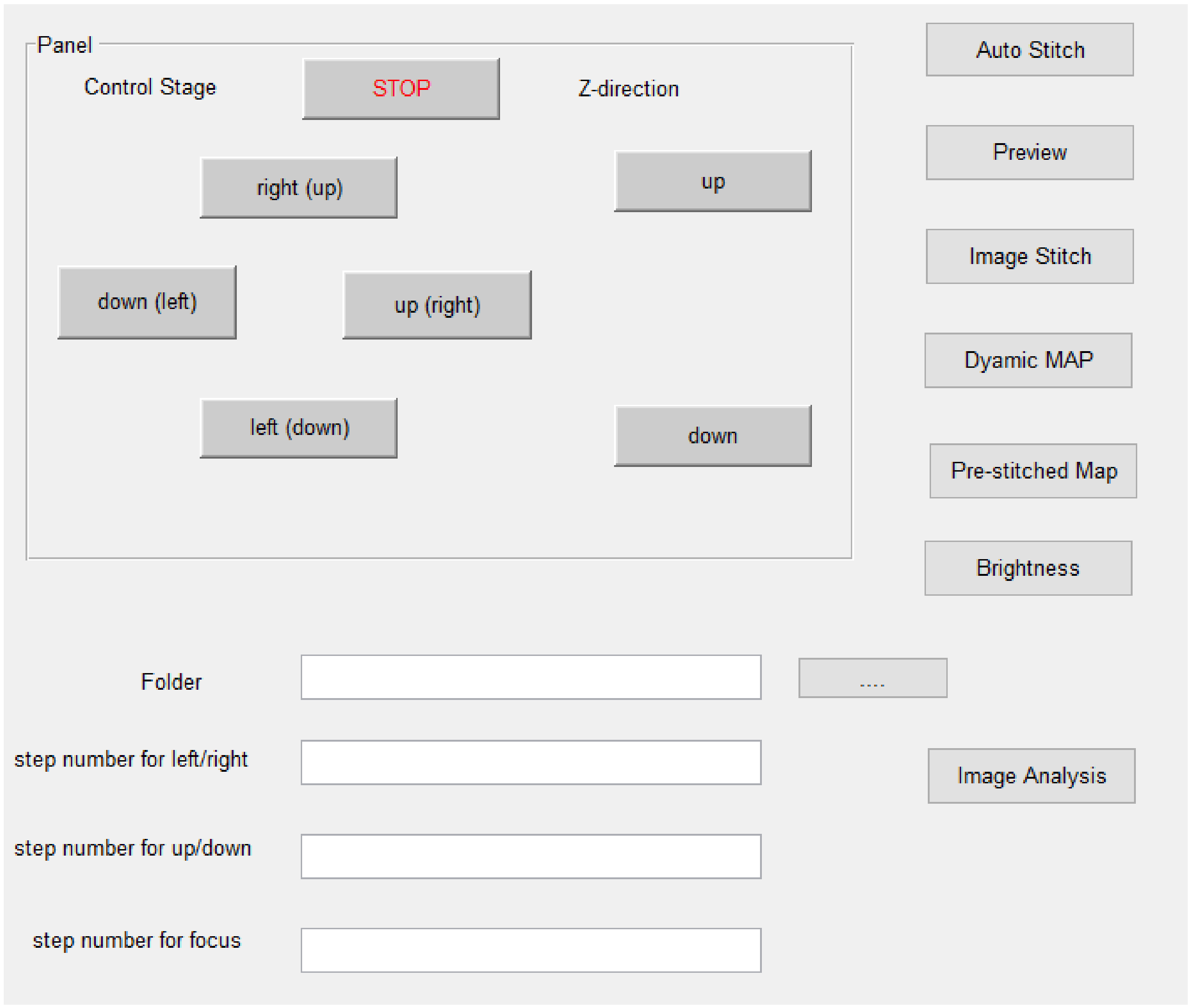


Figure 22: The MATLAB Program GUI.

The upper left region of the GUI is the control panel for the motorized stage. The "Auto Stitch" button will make the motorized stage scan the sample from the current stage location with the shape of a rectangle whose size is X rows and Y columns. The X and Y are user-inputs. The "Preview" button allows a user to move the stage with an image from the camera, manually focus and use "corner stitched" method explained before. The "Image Stitch" button allows a user to stitch any set of microscope images (not necessary from the image acquisition process). The "Dynamic Map" button allows a user to zoom in/out on a stitched microscope image using the dynamic stitching method, which will be explained later. The "Pre-stitched Map" button allows a user to zoom in/out and navigate on a stitched microscope image using the pre-stitched method, which will be explained later. The result of grain boundary detection is viewable and togglable in the last level of the map (A single image with the original resolution). The "Brightness" button adjusts the brightness of the image obtained from the camera. The "Image Analysis" button allows a user to do image analysis automatically on a microscope image (details will be explained in the "Image Analysis Section"). The "Folder" entry is the place where a user can input the location of the microscope image. For example, a user can input the folder that contains the images needed to be stitched. The last three entries allow the user to control the number of steps for the X- and Y-direction of the motorized stage and the Z-direction of objective lenses.

4.2 Image Stitching Software

4.2.1 Modifications to MIST Image Software

The original software, MIST, is developed by the National Institute of Standards and Technology (NIST) [10], [11]. As mentioned in the literature review section, it uses the DFT algorithm to stitch the gray microscope image using only MATLAB language. However, the RGB images need to be stitched together in this project since the images obtained from the microscope are RGB images, and RGB images are easier for the users to see what is going on over the region of interest of the sample. Therefore, the software needs to be improved so that it can take the RGB images and stitch them together.

The improved program will still stitch the grayscale images together first. Then, it will get the coordinates of each image inside the final stitched image and the assembling order of each image. Finally, the new program will stitch the RGB images based on the information acquired in the previous step.

4.3 Multiple Zoom Level Map Design

Before the image stitching process, the program will automatically detect the number of images that will be stitched, and then the program will determine whether it is necessary to compress them. Therefore, if a stitched image shows a large area, very few details can be seen on this stitched image because the images used for stitching are compressed. A user can input the amount of compression for an image. The maximum compression is from the resolution of 2592x1900 to 240x120. Small features are lost during the compression. The only features that can be observed in a large, stitched image are distinctive, such as abnormal grains. To take a closer look at those distinctive features or certain interesting regions on the stitched image a higher resolution image is necessary. There are two ways to design the function of multiple zoom level map features. The next two sections will introduce them and discuss the advantages and disadvantages of each method.

4.3.1 Pre-Stitching Method

The Pre-Stitching Method performs the image stitching ahead of any input from the user to zoom in. The software needs to determine the suitable zoom level for the stitched image and the number of pre-stitched images in each level. Due to the typically large size of the stitched image, we decided to use three zoom levels where the first level is the original stitched image, the second level is a 3 x 3 pre-stitched image with the stride of the size of one image in both row and column direction and the third level is a single image which is directly collected from the camera. Therefore, only the second level needs the pre-stitching process.

The pre-stitching process is time-consuming due to the large size of the stitched image. In this project, a three-level map was created for the stitched image with 60 rows and 47 columns. The second level presents a pre-stitched image whose size is 3 x 3, and the third level is a single image. With the stride of 1 in both the row and column direction, there will be 58 rows of pre-stitched image and 46 columns of pre-stitched image in each row. Therefore, there will be a total of 58*45 = 2610 pre-stitched images. After the experiment, it takes more than 6 hours to finish the pre-stitching work and it takes up more than 40GB for storing them. Another disadvantage is that the size and number of pre-stitched images are fixed, which might result in undesirable zoom-in under certain circumstances.

However, this method is much faster for zooming-in than the Dynamic Stitching Method, which will be introduced in the next section.

4.3.2 Dynamic Stitching Method

In contrast to the Pre-Stitching method, Dynamic Stitching Method will stitch the higher resolution image after the user's input. This means that the program will do the imaged stitching process according to the user's requirements. Therefore, if the input is a rectangle, the program has to use this method because it is impossible to predict what rectangle the user would like to input. If the input is a point in the input image, it will try to determine the best view that can present around this input point. In addition, the number of zoom levels can vary with the size of the zoomed-in region (e.g., the larger zoomed region is, the more zoom level will be).

However, since the stitching process is running in real time, it can take a long time to complete, which makes the dynamic stitching method sometimes very slow. With the same 60 x 47 stitched image used in the previous method, it can take more than 5 minutes to get the zoomed-in region if the point location is at the center of the input stitched image.

4.3.3 Navigation features

Navigation features were added to enable further inspection of the large region of interest. With this feature, a user can navigate around the region of interest with the second zoom level and the third zoom level, in a similar manner to using a microscope to observe the sample with a much larger view than the view of any type of objective lens. If an interesting spot is found, the user can zoom out to a larger region to obtain a better visualization.

4.4 Image Analysis Software

Grain boundary identification is the first step for grain analysis since it is the key step for separating and identifying each grain. Then the PDF, CDF, and grain size map are created based on the previous result for better visualization of grain size statistics. Finally, the technique of locating abnormal grains in a large area of interest is introduced in this section.

4.4.1 Grain boundary identification

As mentioned in the literature review section, image segmentation is used for grain boundary identification. However, instead of inputting the gray microscope image directly to the watershed segmentation method which has been explained before, we improved the algorithm further.

The grain boundaries are visible since they are very similar to the edges and therefore they have a large gradient (the differences between two neighbor pixels on the edges are huge). Hence, after we converted the original image to gray image, we increase the contrast of it to get larger values of those gradients at grain boundaries. This step is done by the built-in MATLAB function imadjust(). This function gets rid of the bottom 1% and the top 1% of all pixel values and then it maps the rest of pixel values to values between 0 and 255. Then we can detect the grain boundaries preliminarily using the gradient information in the enhanced gray image and setting a cutoff threshold. Specifically, we set the threshold to 0.3 and if the gradient value is larger than 0.3, it is a grain boundary. Otherwise, it is a grain. This threshold works for all the objective lenses and materials we have tested. This step is done with the built-in MATLAB function imgradient() and imbinarize(). Since the original image contains lots of noise (e.g., dirt) whose size is usually very small, it might be misidentified as grain boundaries in the previous step. Thus, it is necessary to get rid of the noise by setting another cutoff threshold. We used the built-in MATLAB function bwareaopen() since it can get rid of a cluster whose pixel size is smaller than a threshold. The threshold we used was 500 pixels, which was found to be a good trial and error value. This threshold is valid for all the objective lenses and materials we have tested. For convenience, we will call the result of this step Image α. Then, the watershed method, which is a built-in function in MATLAB, is necessary for closing (completing) those partially detected edges. We first calculate the distance map from Image α using the built-in MATLAB function bwdist(). This map will give the local maximum number to the pixel at the center of each grain and 0 on the grain boundaries. In order to fit the built-in MATLAB function watershed(), we inverted that distant map so that it can turn the local maxima to local minima and place the water source at the center of each grain (local minimum). Due to potential over segmentation issues, we also used the built-in MATLAB function imhmin() which suppresses all the pixel distances that are shorter than the threshold set by the user. The threshold we used is 5 based on the experiments. This threshold will vary with different objective lenses and different types of materials for better grain boundary detection. Finally, we dilated all the grain boundaries detected using the MATLAB function imdilate() in order to make them visible. Figure 23 shows the results of each step used for grain boundary identification, and Figure 24 shows the resulting image, which is the binary image after dilation.

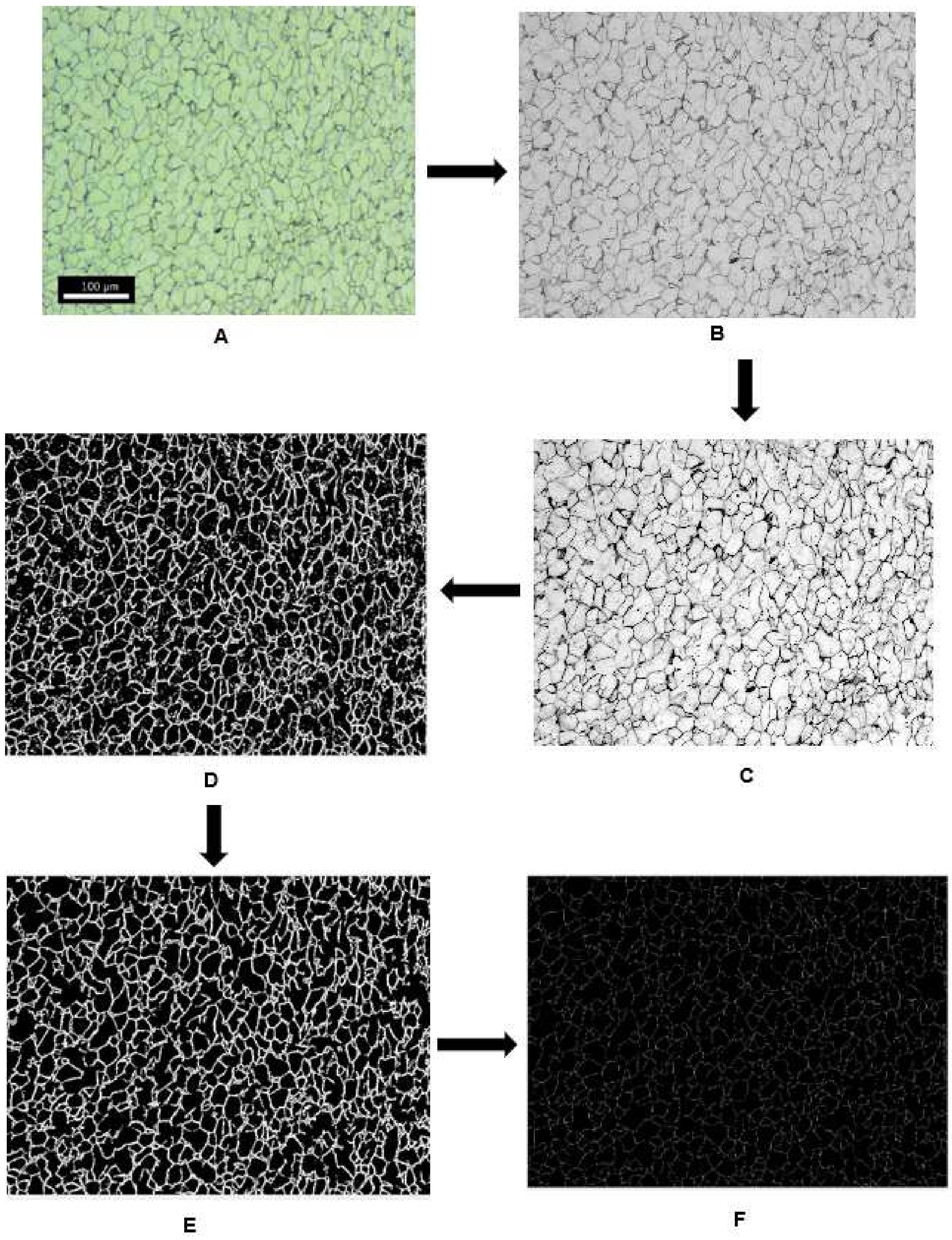


Figure 23: Grain boundary identification process. (A) Original image before processing. (B) Grayscale image of the original image A. (C) Enhanced grayscale image using imadjust() function

on grayscale image B. (D) Binary image after finding the gradient and applying the 0.3 cutoff threshold to delineate grains and grain boundaries. (E) Binary image after applying noise filter. (F) Binary image after applying watershed. The grain boundaries are 1 pixel wide.

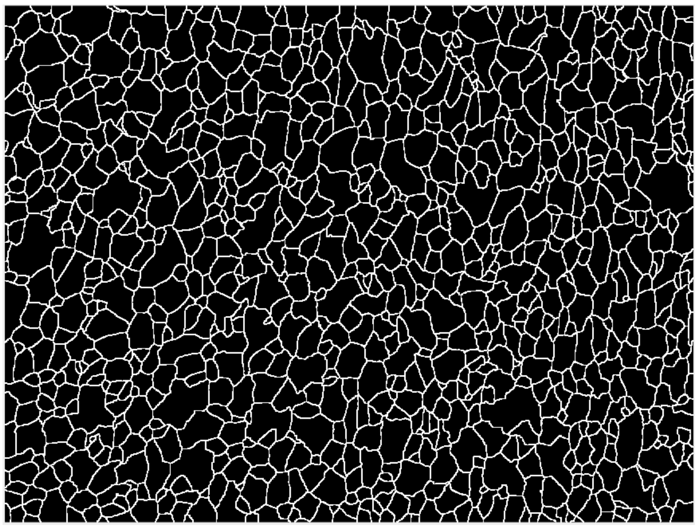


Figure 24: Final result image. Binary image after dilating grain boundaries to 5-pixel width.

4.4.2 Grain boundaries stitching

It is inefficient and inaccurate to identify the grain boundaries on a stitched image based on several experiments that we did with the current grain boundaries identification program. This is caused by the large size of a stitched image. Therefore, we figured out a novel way to identify the grain boundaries in a large image, which is first identifying grain boundaries in a single image, the one that will be stitched, and then stitching all the resulting images together (e.g. instead of stitching all the original images collected from the camera together, stitching all the grain boundaries together).

There are two ways to stitch the grain boundaries. One is to use the same stitching method as the one to stitch the original image. The other one is to use the same locations and assembling order as the ones used for the same set of original images.

Although both methods work well, there are differences between these two methods. For example, Figure 25 and Figure 26 show the same spot on a sample but has different grain boundaries. This happens because the order of assembling is different. Figure 27 shows an image that contains the region marked in Figure 25 and Figure 26. Obviously, the direct stitching method covers the marked feature in Figure 27 by another image while the other method does not.

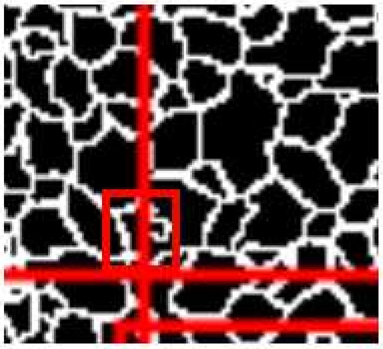


Figure 25: The result stitched image using the same locations and assembling order. The feature marked in Figure 27 appears in the small red rectangle.

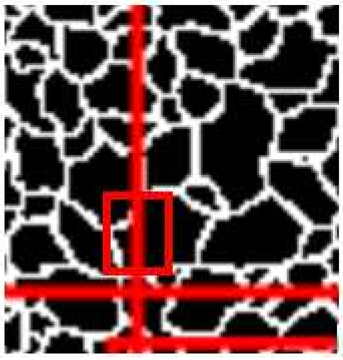


Figure 26: The result stitched image using direct stitching method. The feature marked in Figure 27 disappears in the small red rectangle since it is covered by another image that does not contain this feature.

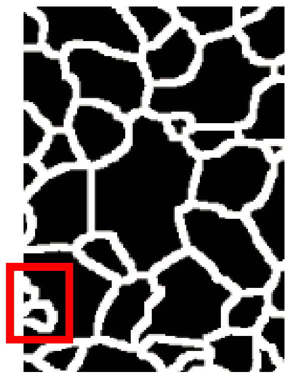


Figure 27: Original grain boundary image showing the feature marked in the red rectangle.

4.4.3 Grain Size Analysis

This section describes how to get the grain size and make a probability density function (PDF) plot and cumulative distribution function (CDF) plot for the grain size.

In order to get number-weighted PDF (called PDF plot in the rest of paragraph) and CDF plot, all the grains in a microscope image have to be measured. My MATLAB program is designed to have two ways for implementing the measurement of the grain size, which will be elaborated on in the 1next two sub-sections. A CDF plot can be created via the MATLAB built-in function CDFplot() directly after obtaining all the grain size statics. However, there is no function in MATLAB that can be directly used for a PDF plot. Therefore, we developed our own function for generating a PDF plot. First, all the grain size data needs to be divided into X ranges, where X is a user-input parameter, using the MATLAB built-in function hist(). This function will divide the data into X ranges (from low to high) and count how many data point appears in each range. Then the entire area under the PDF plot is calculated by the MATLAB built-in function trapz(), which approximates the area under a discrete function by breaking the area down into trapezoids and adding them all together [22]. Finally, the probability of each range can be calculated by dividing the number of grains in each range by the entire area calculated before. Then the PDF plot can be generated based on all the ranges and probability of them. A curve can be added into the PDF plot if this curve if fitted into the histogram. An example of a number-weighted PDF and CDF plot is shown in Figure 28 and Figure 29.

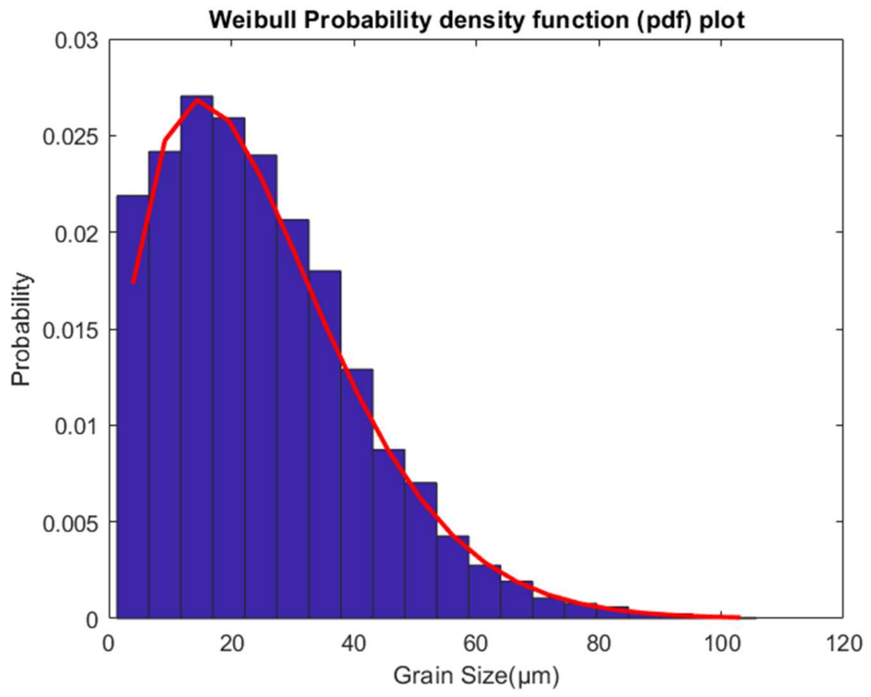


Figure 28: The number-weighted PDF plot of grain size distribution for the shaft part of steel bolt shown in Figure 23 (A). The grain size is measured via Linear-intercept Method. This PDF plot has 20 ranges (X=20). The red curve fitted for PDF plot is a Weibull curve.

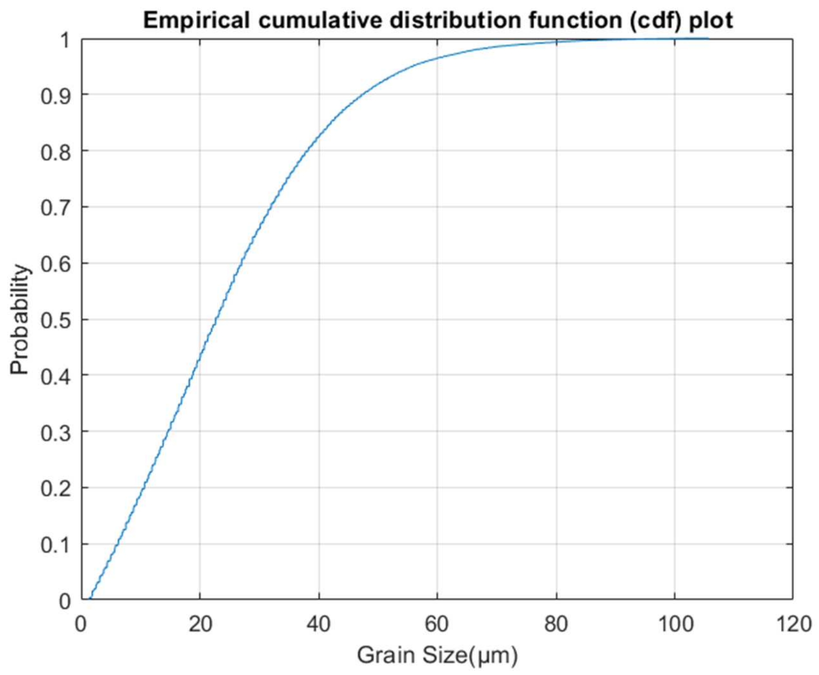


Figure 29: The CDF plot of grain size distribution for the shaft part of the steel bolt shown in Figure 23 (A). The grain size is measured via Linear-intercept Method.

An area-weighted PDF plot is used for a large region in the ceramic since a bimodal distribution for the grain size is expected for this ceramic sample and an area-weighted PDF plot is more sensitive to the large grains. The method for creating an area-weighted PDF plot is very similar to the one of creating a number-weighted PDF plot. The only difference is that instead of dividing the number of grains in each range of the grain size by the entire area under the PDF plot, the program divides the total area of grains of each range by the total area of all the grains.

4.4.4 Linear-intercept Method

Instead of measuring the length of the line that crosses the grains in an area as defined in ASTM E112 [16], the program uses a simpler version of the linear-intercept method [23], [24] to measure the grain size. Typically, there are four types of lines at 4 angles (0°, 45°, 90°, 135°). Each type of line is repeated multiple times on the image, at a spacing of 5 pixels. Each type of line will cover all the images, and each line is only 5 pixels apart. For example, if the horizontal direction (0°) is measured, the line starts from the first row and the next line is 5 pixels off from the previous line. This process continues until the line reaches the last row of the image. A user can change the space between the lines for accuracy or efficiency. Then, all the lengths of segments in each grain will be recorded into a variable named grain_size_hor for example. In order to implement this method, I used the MATLAB code written by Lhteo [23], [24]. Although this program has image segmentation built-in (i.e., it can convert a grayscale image to binary image), this built-in feature was bypassed by inputting binary images that were produced according to the algorithm shown in Figure 23 and Figure 24. After inputting the binary image (grain boundary image), it will start the process of measuring the grain size. While measuring, MATLAB will consider measurements smaller than three pixels as noise and thus exclude them. Results from each measurement direction are saved into their own variables, named grain_size_hor, _ver, _45_1, _45_2. The users are able to define their own grain size based on these four variables. For this thesis project, we defined the grain size as the average of these four variables.

4.4.5 Pixel-count Method

Using the equivalent circular area diameter is a much simpler way to measure the grain size than the linear-intercept method. The program counts how many pixels are inside each grain and converts them back to the real size in μm^2 according to the real area size of each pixel in the image. After the actual size of the grain is known, an imaginary circle that has the same area of the grain will be used (as shown in Figure 30) to find the grain diameter. The grain diameter can be calculated as

$$D_a = 2\sqrt{\frac{A}{\pi}}$$

Where A is the actual size of the grain and D_a is the grain diameter. This method runs much faster than the previous one.

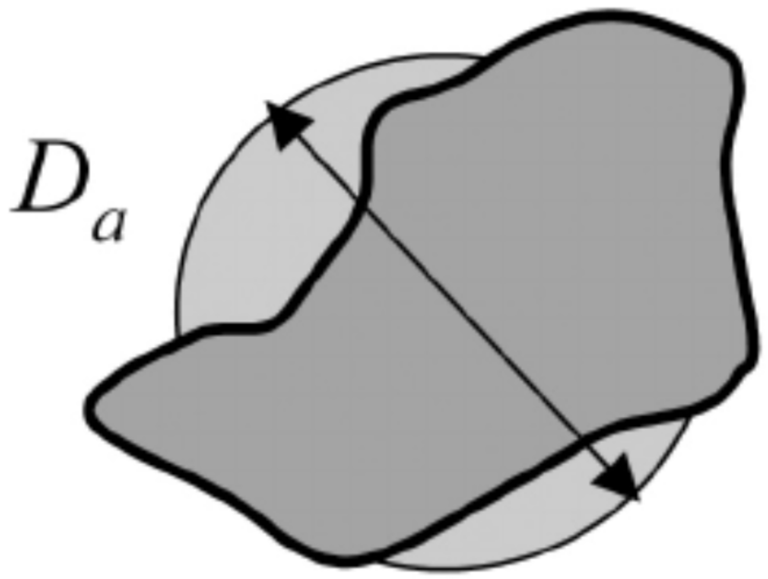


Figure 30: An imaginary circle that has the same area of the grain. The grain size is defined as the equivalent diameter of a circle, D_a, with the same area as the grain.

4.4.6 Grain Size Map

In order to visualize the grain size in the original image clearly, the grain size map is generated, and a color bar is used for better visualization. In the grain size map, red indicates the large grains, while blue indicates the small grains. Therefore, a user can see the relationship between the location and grain size clearly from the grain size map.

Various units for the grain parameter and different methods to measure the grain size will yield a different grain size map. In this project, we generated a grain size map with grain diameter and the pixel-counted method is used for the grain size measurement. An example map is shown in Figure 31. It is also possible to create a grain size map for a stitched image since all the grain size data can be obtained by the program. However, it requires a large amount of Random Access Memory to create a large grain size map. Therefore, it is impossible to generate such grain size maps on a personal laptop due to the limitation of memory. A trial has been conducted on a supercomputer, but due to the low-level graphic errors that we cannot solve on the supercomputer, we are unable to get a grain size map for the stitched image.

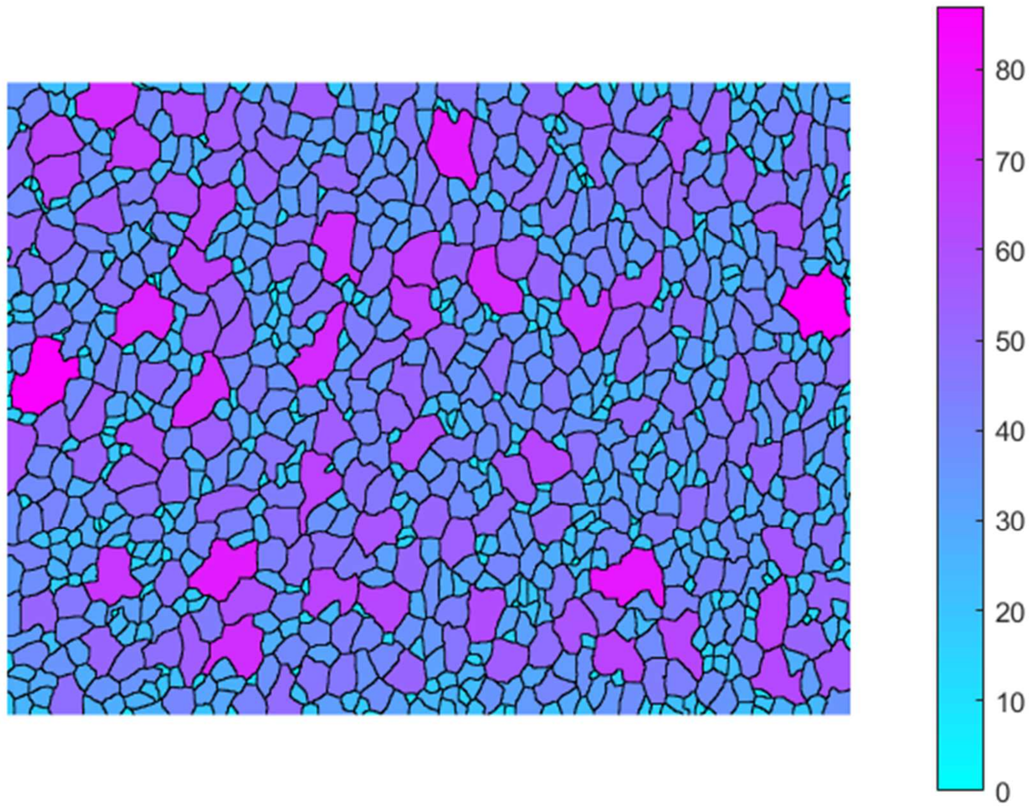


Figure 31: The grain size map for an original image obtained from the shaft of the steel bolt sample shown in Figure 23 (A). The number beside the color bar is the grain diameter in μm measured via the pixel-counted method. The color bar also indicates that larger the grain diameter is, more purple it is; smaller the grain diameter is, more bright blue it is.

4.4.7 Aspect Ratio Analysis & Map

There are several ways to define the aspect ratio for a grain. In this project, we used the minimum bounding box method to find the aspect ratio, because it is simple to implement in the software. MATLAB has a function [25] that can find the minimum bounding box for each grain as long as the grain boundary is provided. After the minimum bounding box is found, the aspect ratio can be calculated by dividing the longer side of the minimum bounding box by the shorter side.

Similar to the grain size map, aspect ratio map is generated for better visualization and to show the relationship between the location and aspect ratio. Figure 32 shows an example map.

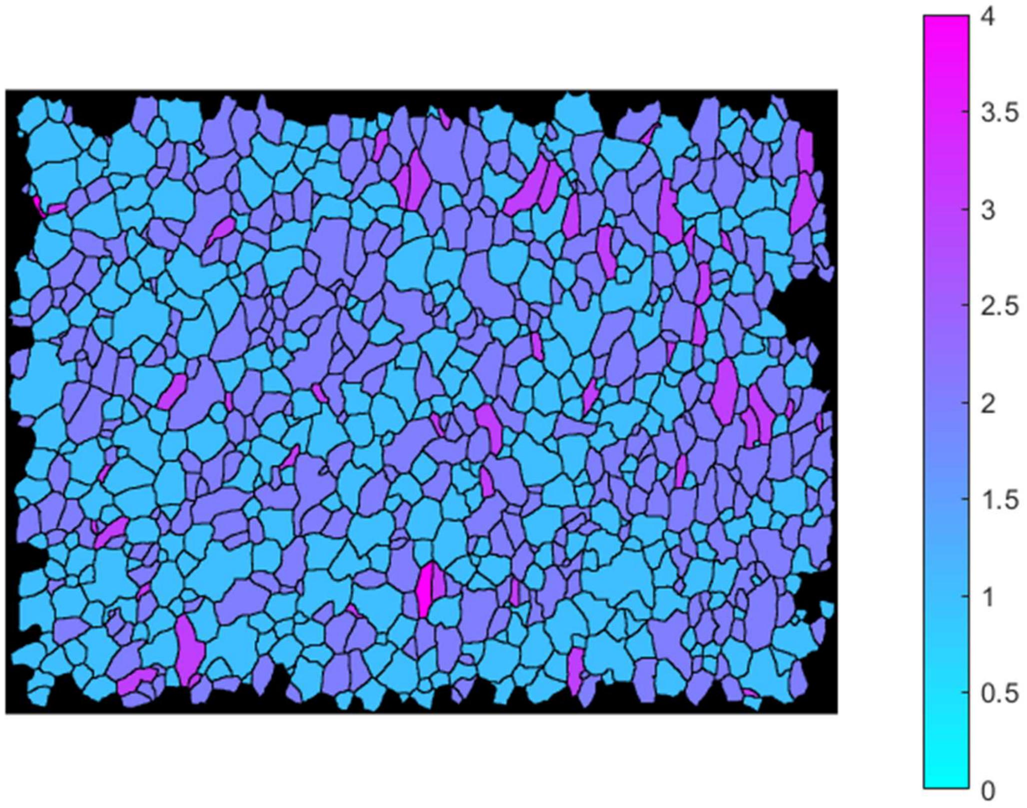


Figure 32: The aspect ratio map for an original image obtained from the shaft of the steel bolt sample shown in Figure 23 (A). The number beside the color bar is the aspect ratio. The color bar also indicates that the larger the aspect ratio is, the more purple it is; the smaller the aspect ratio is, the more bright blue it is.

4.5 Locating Large Grains

The rare event we tried to find in this project is to locate abnormally large grains over a large region interested. Specifically, we wanted to locate the largest X grains, where X is a user-defined number. The pixel-count method is used due to its efficiency.

By inputting a stitched image, the program can identify individual grains and determine the size of each of them using the pixel-count method. Then, based on the ranking of the grain size, the program can flag them for visualization. We designed two different ways to format the results so that a user can examine the same output with different viewing.

Colormap

There is a MATLAB built-in function called colormap() that can flag the largest X grains. However, there is a problem with this function. It will make the size of the flagged map different from the size of the original image and there will be gray borders around the image. Therefore, there is no way to flag the largest grains on the original image based on the colormap. Figure 33 shows an example result, and Table 2 shows the time needed when using this method.

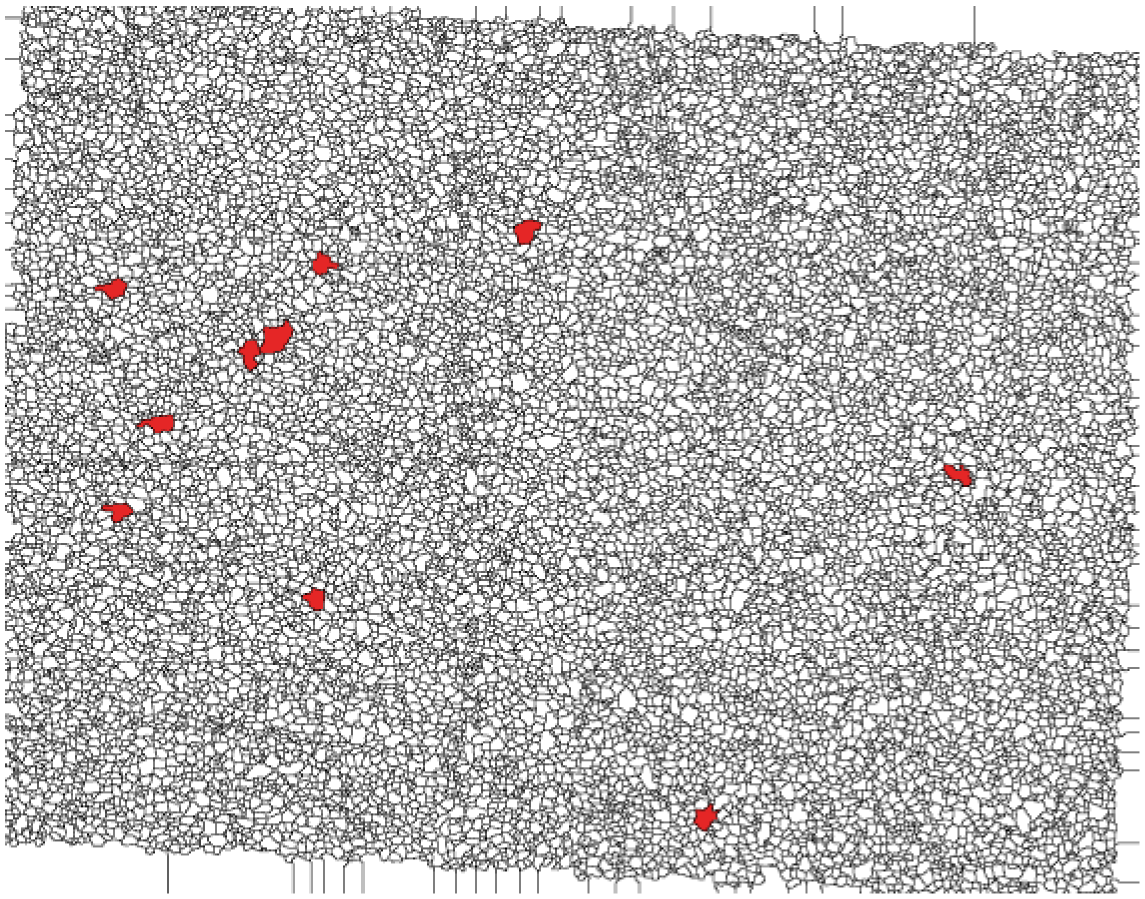


Figure 33: The result grain size colormap for a stitched image, which contains 7 rows and 7 columns, obtained from the shaft of a steel bolt sample shown. The red grains are the 10 largest grains on the original image. The black lines around the borders are the noises and should be ignored since there is no grain sized information in the white area on the borders.

Table 2: Time needed to find 10 largest grains using the colormap method. The percentage of total time spent on each step is shown in parentheses.

Type	Find Grain Size Time	Find 10 largest grain size time	Image Creating Time	Total Time
Single Image	2.5 second (88.03%)	0.046 second (1.62%)	0.294 second (10.35%)	2.84 second
7x7 Stitched Image	456.98 second (98.65%)	3.53 second (0.76%)	2.72 second (0.59%)	463.23 second

4.5.1 Flag on the original Image

Since it takes too much time to use the previous method, we decided that instead of creating a map, directly flagging on the original image might be a good method. This method helps to determine whether the flagging is correct or not. Figure 34 shows an example.

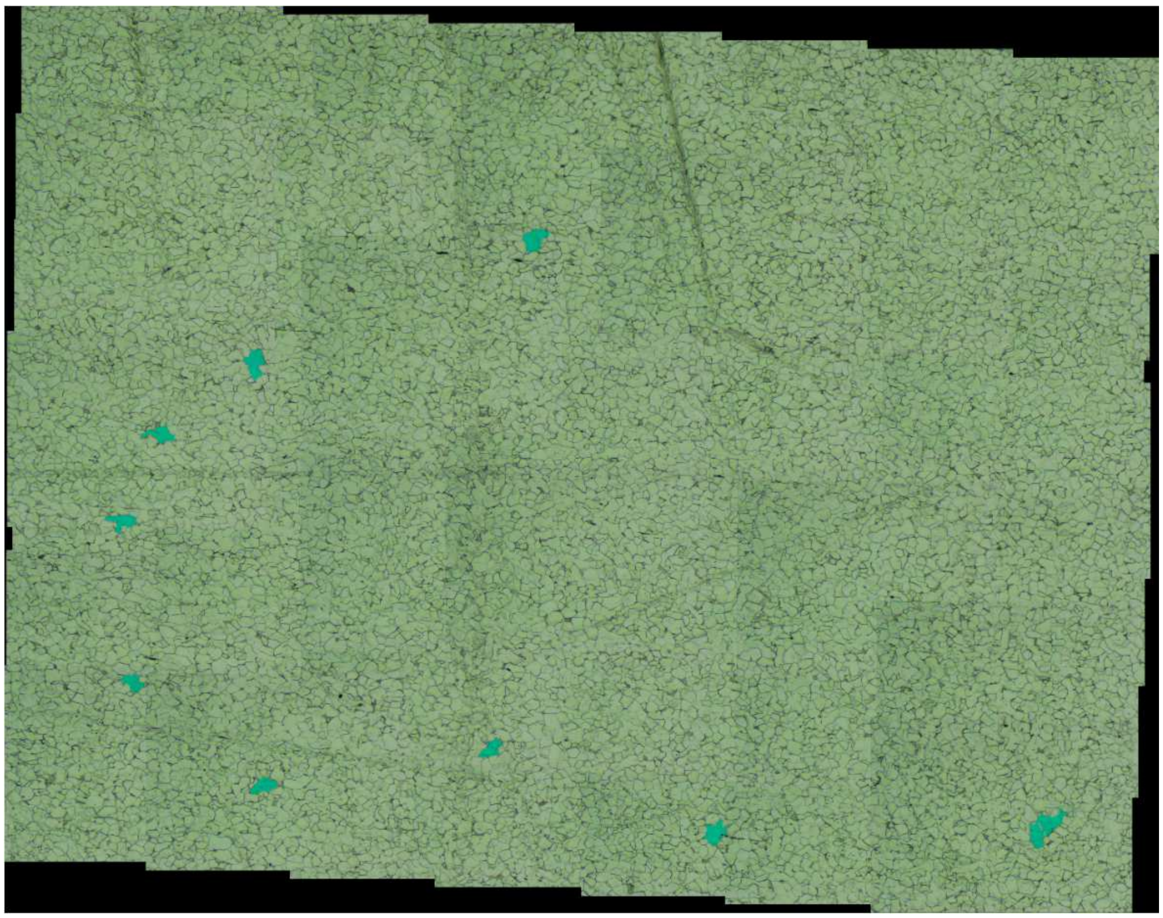


Figure 34: The largest 10 grains are directly flagged on the original image which is a stitched image with 7 rows and 7 columns. The light green grains are the 10 largest grains on the original image. The original image is obtained from the shaft of a steel bolt sample.

As shown in Table 3, this method is significantly faster than the approach for creating a map. Its speed is comparable to the colormap approach.

Table 3: Time needed for flagging 10 largest grains on the original image. The percentage of total time spent on each step is shown in parentheses.

Type	Find Grain Size Time	Find 10 largest grain size time	Image Creating Time	Total Time
Single Image	2.17 second (54.5%)	0.049 second (1.2%)	1.76 second (44.3%)	3.98 second
7x7 Stitched Image	455.4 second (98.65%)	4.4 second (0.95%)	1.85 second (0.4%)	461.65 second

4.5.2 Speedup for abnormal grains location

As we can see in all the tables above, determining the size of all of the grains can take a long time when the image is large. This section discusses a method that avoids finding all the grain sizes but still locates the largest X grains.

MATLAB differentiates each grain by assigning them different numbers in a matrix. For example, all pixels in the first grain will have the number "1". If this grain is made up of 10,000 pixels, the number "1" will appear 10,000 times. Therefore, we can find the largest grain by finding the number in the matrix that appears most frequently. After we find the most frequent number, we record it and discard it from the matrix so that we can continue to find the second most frequent number and so on. There is a MATLAB built-in function mode() that can help to find the most frequent number in a matrix.

Hence, instead of doing ranking for all the grain size, the program will use the MATLAB built-in function mode(), which directly finds the largest grains with the pixel-count method from the grain boundaries map without finding all the grain sizes. With the example of a 7x7 Stitched

Image, this method reduces the total time from 461.65 to 61.06 seconds (shown in the last entry in Table 3) for locating the 10 largest grains and flagging them on the original stitched image.

5. RESULTS

First, we tested the model with a region with a size larger than 1-inch x 1-inch on a metal sample, a grade 2 steel bolt, using a 20X objective lens. It took about 4 hours and 28 minutes to finish getting the stitched image of it, and the actual size of the entire cross-sectional area of the sample is 2.74cm x 2.15cm. In the resulting stitched image, each row contains 60 images, and each column contains 47 images. Therefore, there are a total of 2820 images used to stitch the image shown in Figure 35. The specific time for each part of the process is shown below:

- X-Y movement time: 22 minutes and 16.2 seconds
- Auto-focus time: 161 minutes and 4.4 seconds (average 3.42s per image)
- Image saving time: 70 minutes and 30 seconds
- Image stitching time: 13 minutes and 53.4 seconds

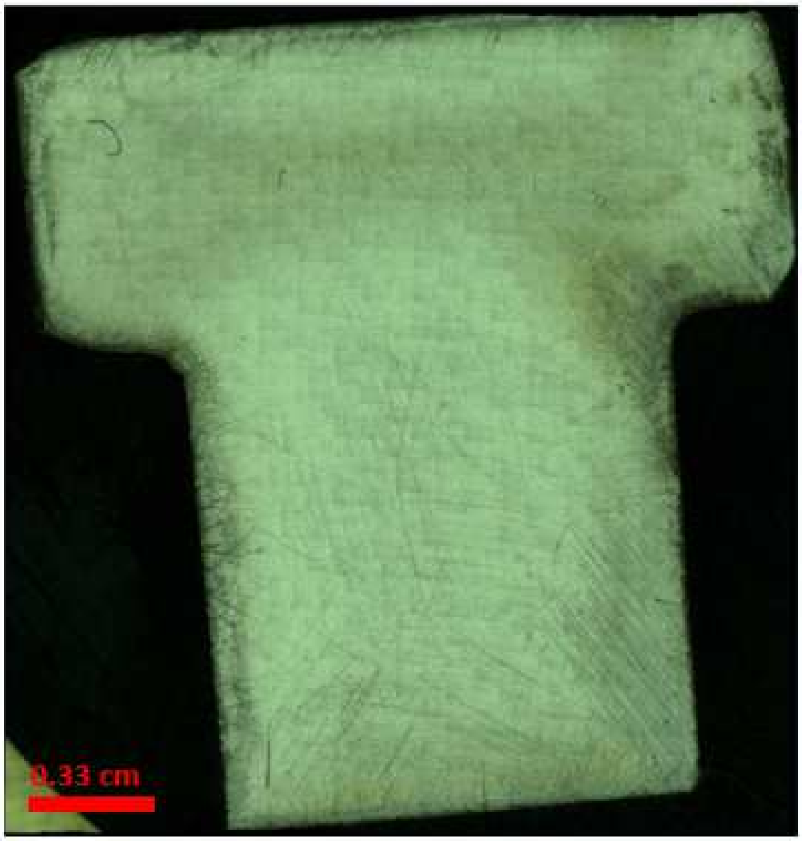


Figure 35: A stitched image (20X objective lens) from the cross-section area of a steel bolt and it contains 2820 images with 60 rows and 47 rows.

To further test the robustness of the model, we tested it with a grade 2 steel bolt annealed for one week using a 20X objective lens. Due to the annealing, many abnormal grains appeared, and they made the entire region have very different features (i.e., large differences in sharpness). The resultant stitched image is shown in Figure 36, and it took about 3 hours and 36 minutes to finish. The actual size is 2.5cm x 1.6cm. In the resulting stitched image, each row contains 55 images and each column contains 35 images. Therefore, there are a total of 1925 images.

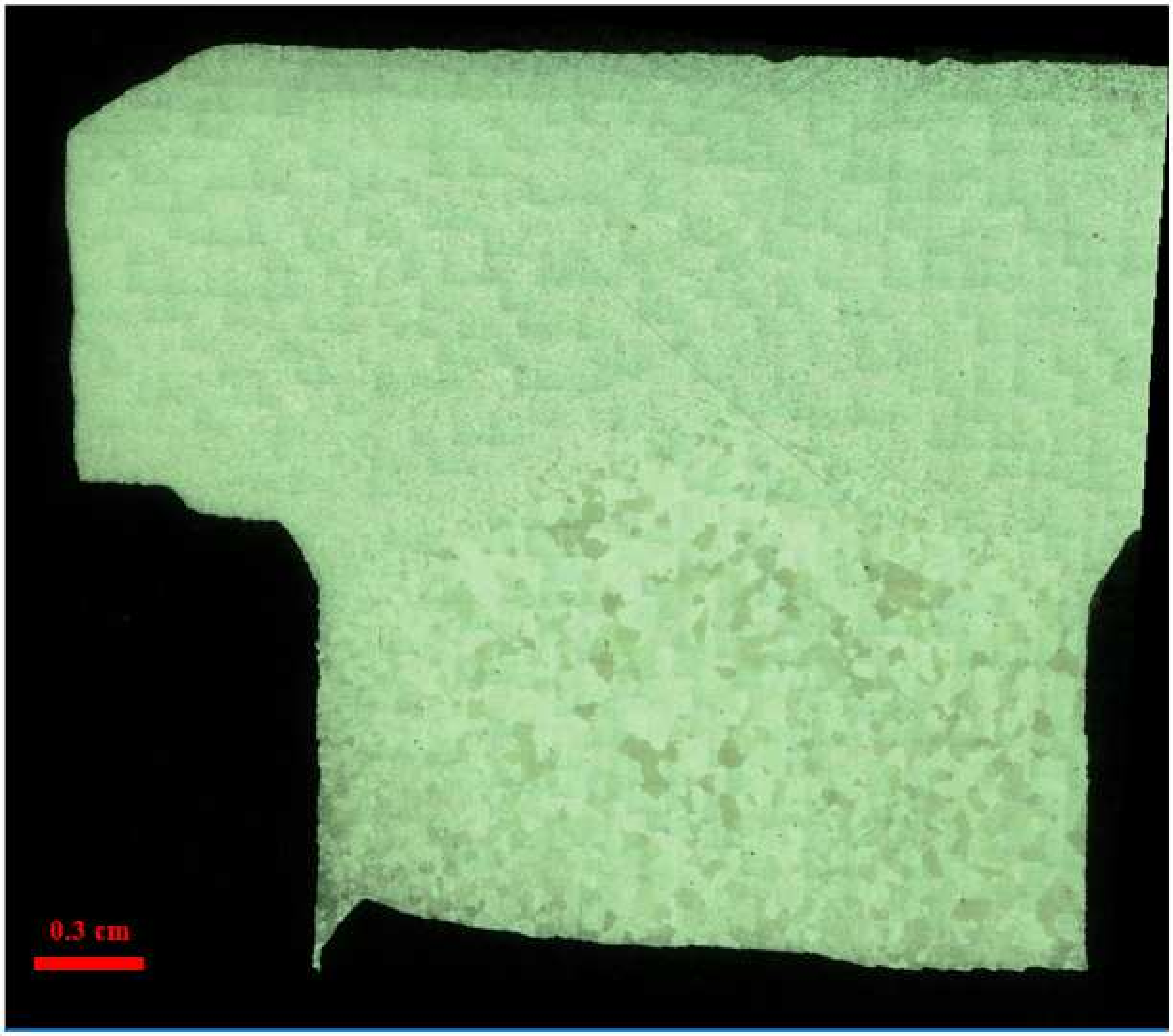


Figure 36: A stitched image (20X objective lens) from the entire area of a grade 2 steel bolt annealed for a week long and it contains 1925 images with 55 rows and 35 rows.

Then, we tested the model with a non-metal, ceramic specimen, using a 20X objective lens. Since the entire region of the ceramic sample was much smaller than the metal one, it only took about 52 minutes to finish scanning and stitching. The actual size of region scanned is 0.78 cm x 0.73 cm. In the resulting stitched image, each row contains 17 images and each column contains 16 images. Therefore, there are a total of 272 images to stitch the image shown in Figure 37.

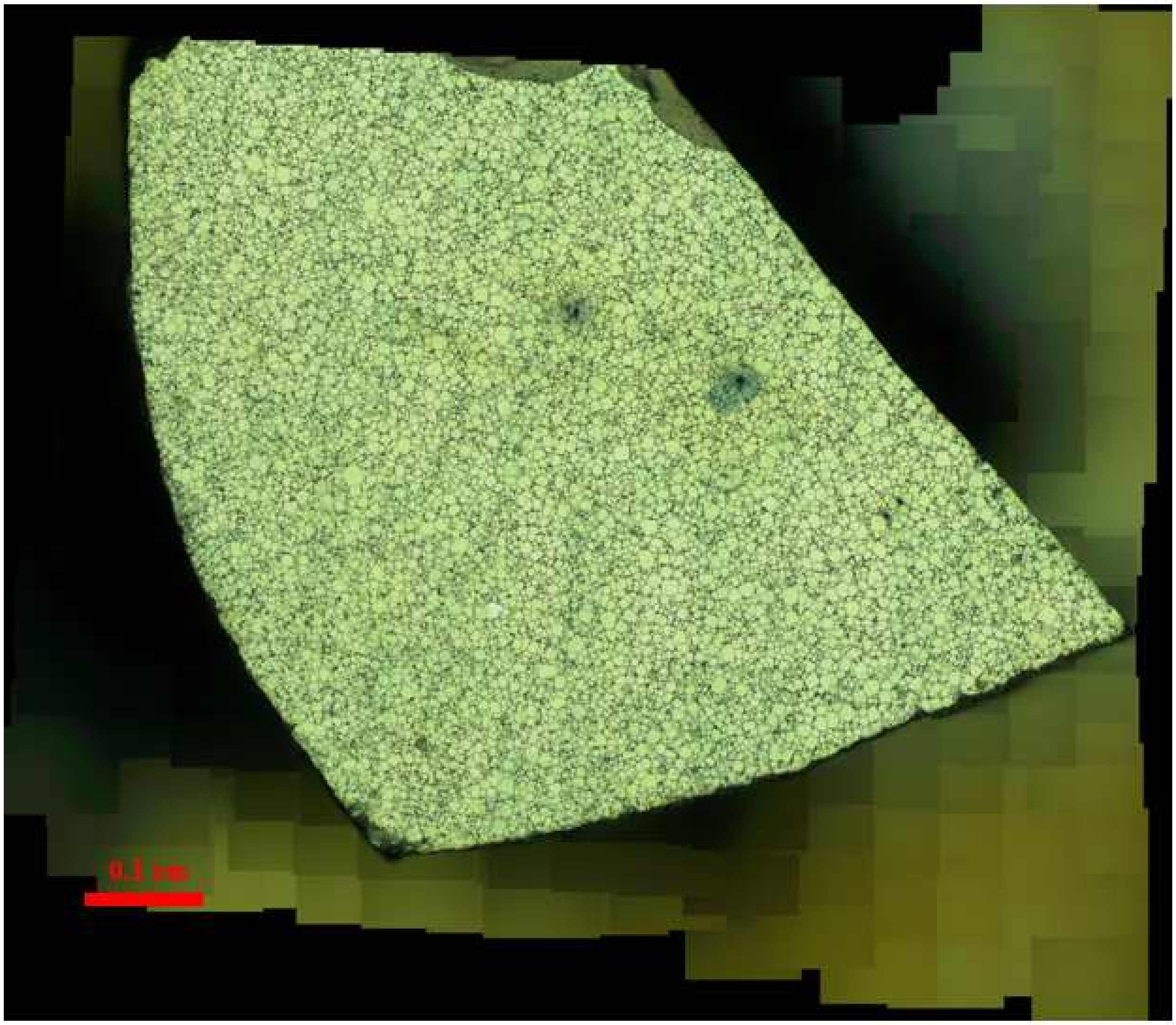


Figure 37: A stitched image (20X objective lens) from the entire area of a ceramic sample and it contains 1836 images with 17 rows and 16 rows.

Finally, we changed the objective lens from 20X to 100X to ensure the robustness of the model. Using the same ceramic specimen as before, we scanned a large area and it took about three hours and nine minutes to obtain the stitched image. In the resulting stitched image, each row contains 51 images and each column contains 36 images. Therefore, there are a total of 1836 images to stitch the image shown in Figure 38.

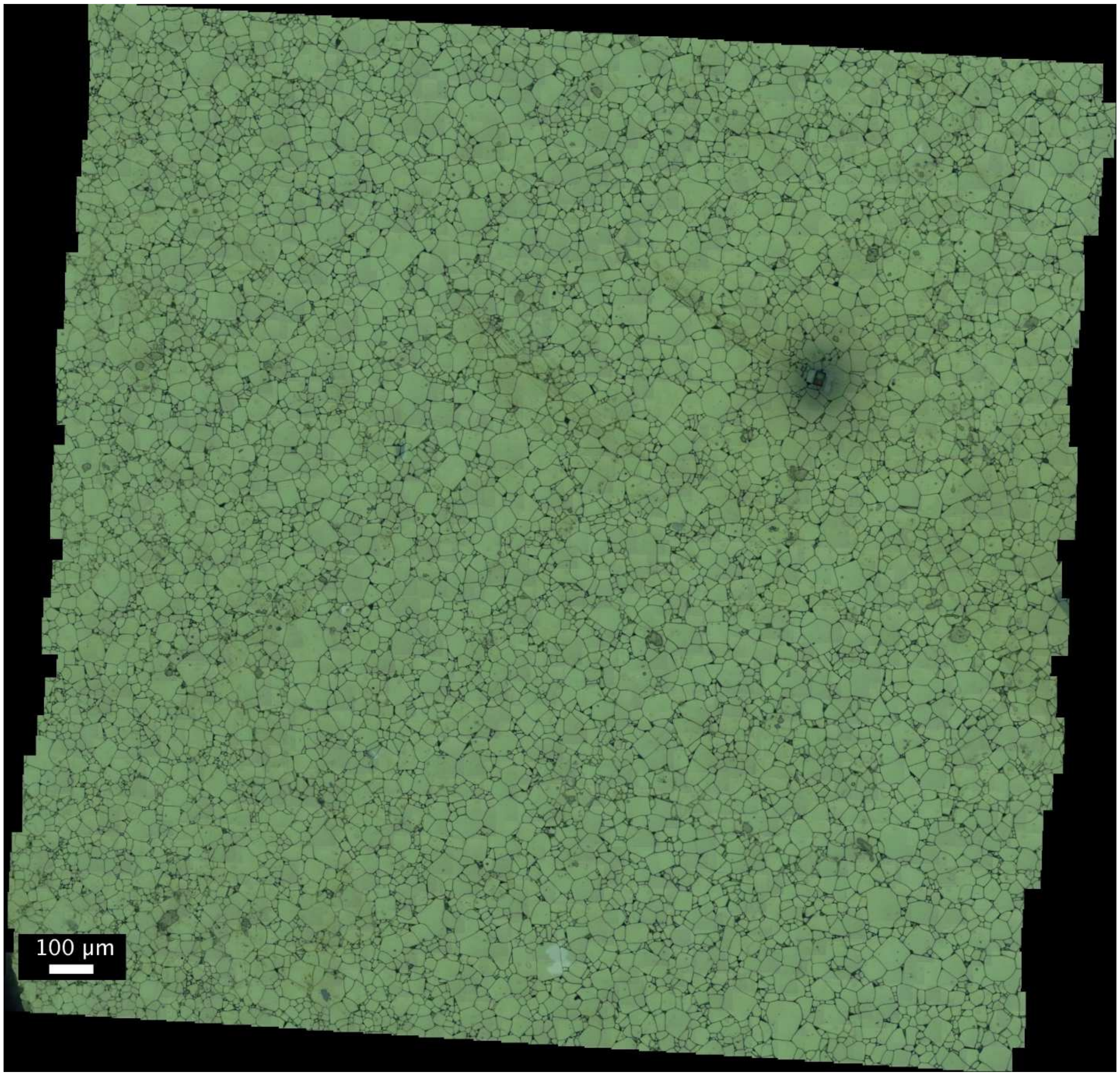


Figure 38: A stitched image (100X objective lens) from a large area of a ceramic sample and it contains 1836 images with 51 rows and 36 rows.

6. DISCUSSION

6.1 Validation for grain size statics

Figure 23 (A) is used for validation of the program that measures average grain size automatically. From the Linear-Intercept Method implemented by the MATLAB program introduced before, the average grain size is 25.3 µm. From the pixel-count method, the average grain size is 31 µm for the same image. The process of manual linear intercept calculation of the average grain size is shown in Figure 39 and the result is 29.2 µm. Since our program uses the pixel-count method for grain size analysis, the results of manual measurement and auto measurement are close to each other, and the difference between them is only 5.8%.

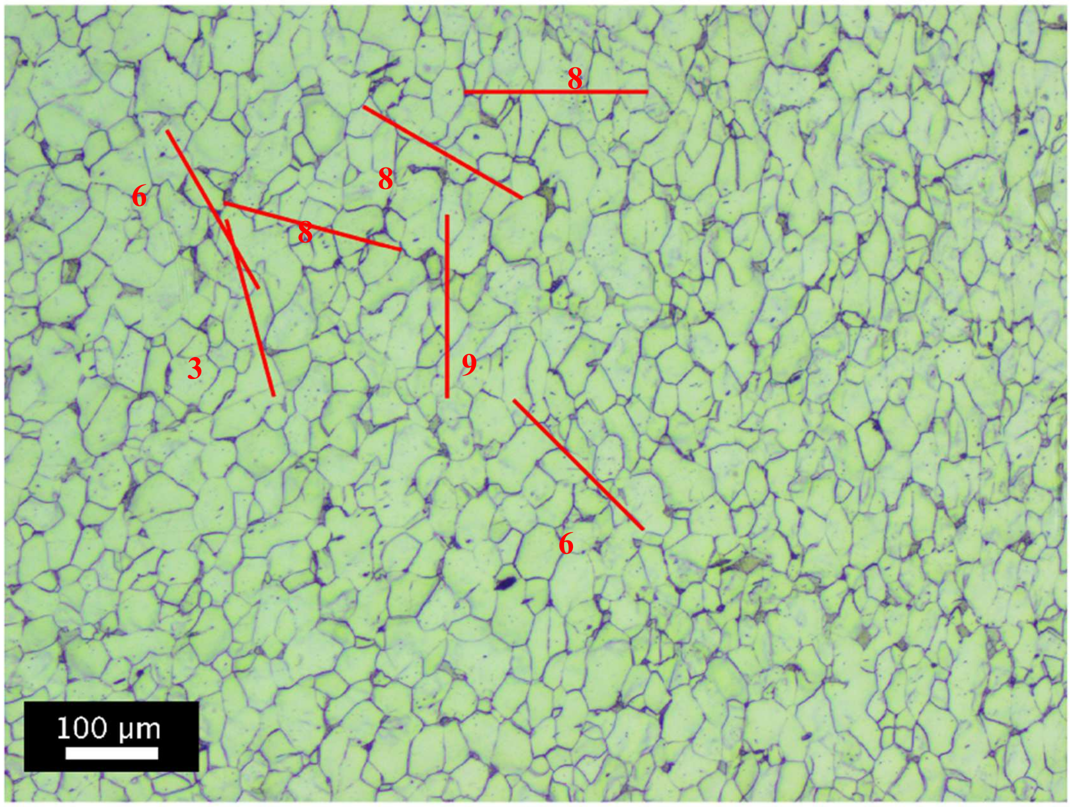


Figure 39: Diagram of manual measuring for Figure 23 (A). The actual length of the scale bar is 16mm as it appears on the paper. Each red line is measured and found to be 32mm. The number near each line represents the number of grain-boundary intersections.

6.2 Grain Boundary Identification Validation

To verify the correctness of the grain boundary identification program the identified grain boundary is applied to the original image. From Figure 40 and Figure 41, the result is reasonable except for some errors from the etching process and dirt on the microscope objective lenses, which will be elaborated in the limitation section.

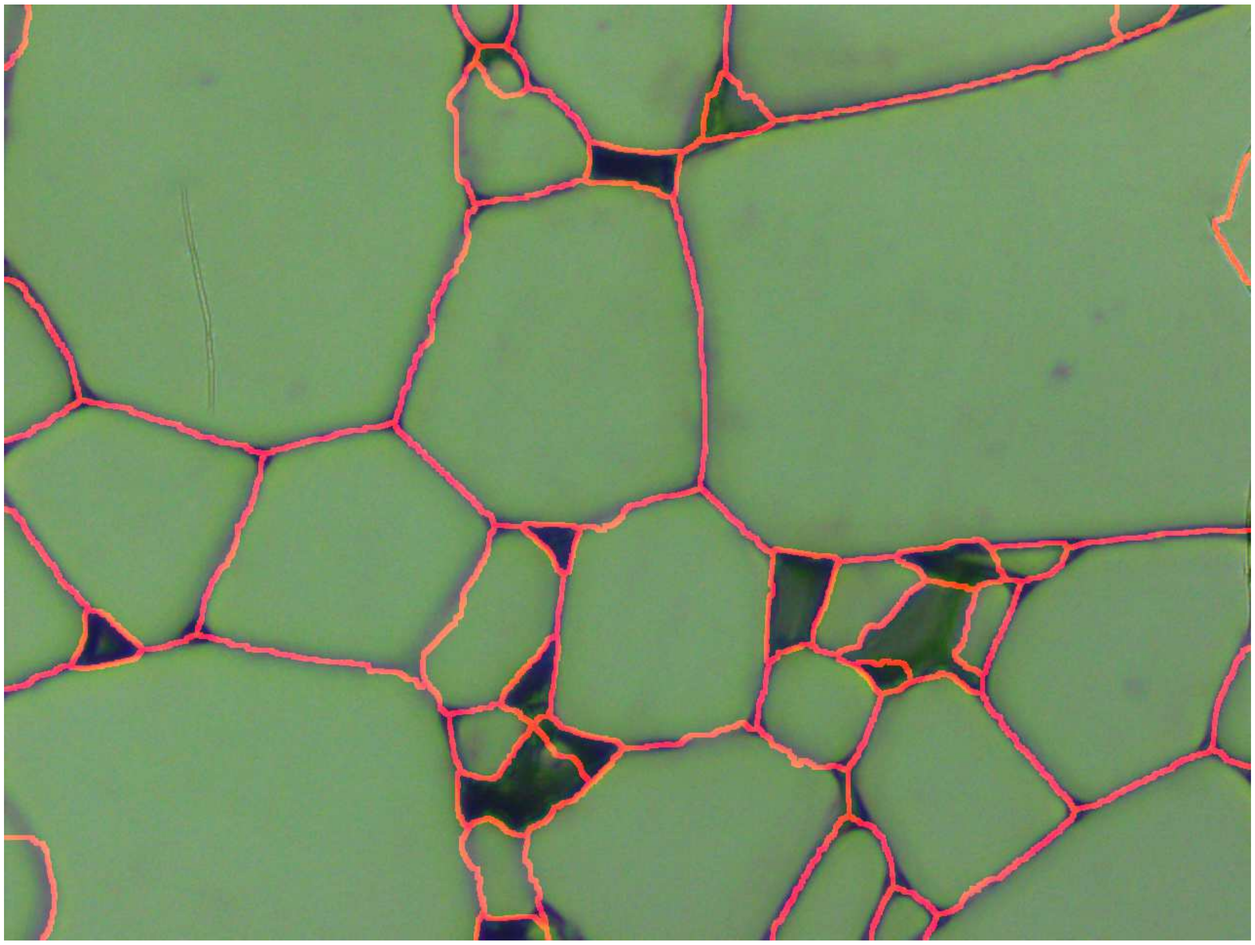


Figure 40: An image from a ceramic sample using 100X objective lens overlaid with the grain boundary identification result

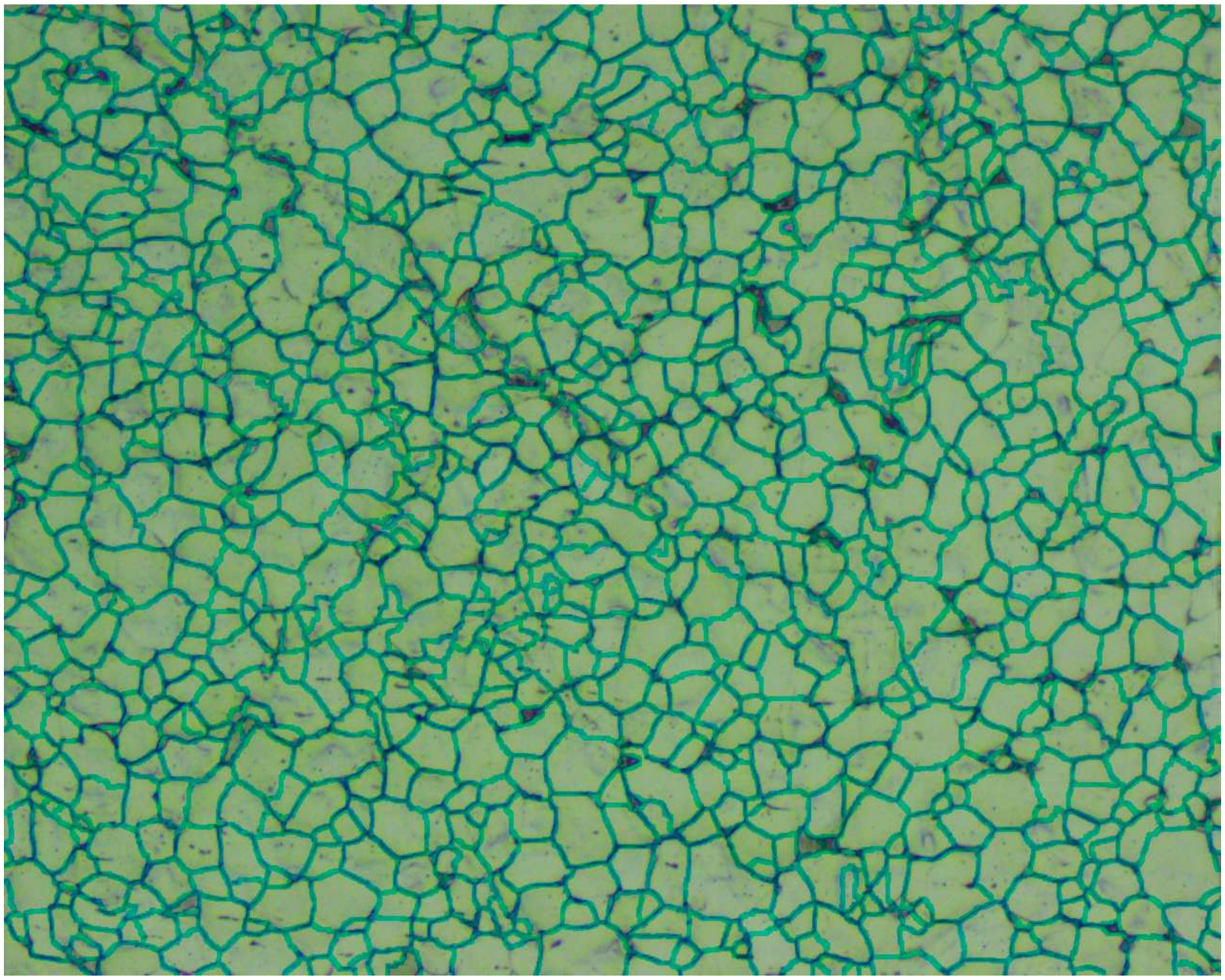
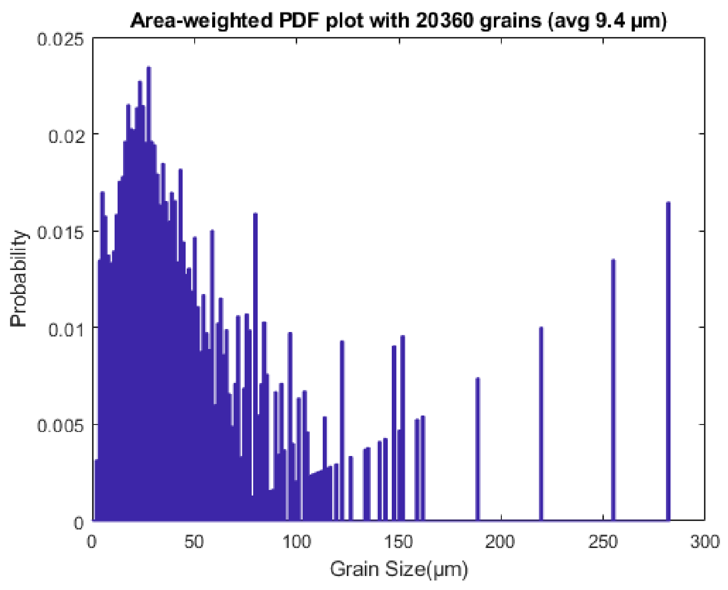


Figure 41: An image from a grade 2 steel bolt using 20X objective lens overlaid with the grain boundary identification result

6.3 Analysis on an individual image and stitched image

There are two ways to analyze all the grains on a large stitched image. One is to do the analysis on all the grains directly on a stitched image. However, due to the large size of Figure 38, it is hard to implement this method on a local computer, which typically is a personal laptop, due to the limitation of memory. A supercomputer, which is a computer with large Random Access Memory (RAM) and several GPUs and CPUs, can be used with this approach. The other method is to divide a stitched image into sub-images to do the analysis and combine all the results of the sub-images together. An area-weighted PDF plot from this method is shown in the left side of Figure 42. and it is compared to the result from a supercomputer, which is shown in the right side of Figure 42.

As expected, the result from the local computer has more grains than the result from the supercomputer due to the overlapping issues. It means that each sub-image will have a little overlapping area that results in the fact that the grains in the overlapping region are counted more than one time.



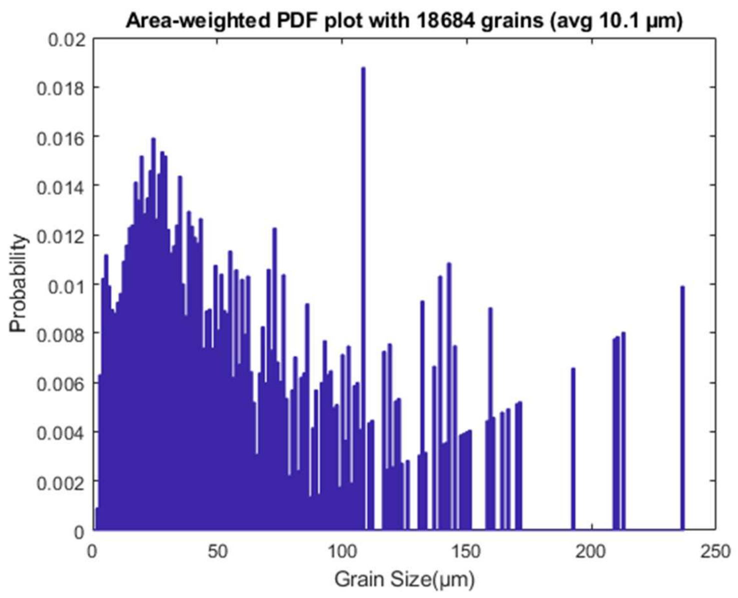


Figure 42: Area-weighted PDF computer for a large region of the ceramic sample shown in Figure 38. The left figure is from a local computer while the right figure is from a supercomputer. The grain size is measured via the pixel-count method. Both PDF plots have 200 ranges (X=200).

In addition, there is an artificial trend at the right part of area-weighted PDF. It is caused by the error of the grain boundary identification program and grain boundary stitching process, which will be discussed in the limitation section.

6.4 Number-weighted PDF v.s. Area-weighted PDF

The number-weighted PDF plot for the region of the ceramic sample shown in Figure 38 is also created to do the comparison with the area-weighted PDF plot. The result from the local computer is shown in the left side of Figure 43 and the result from the supercomputer is shown in the right side of Figure 43.

Instead of the bimodal distribution shown in the area-weighted PDF plot, the number-weighted PDF plot shows a unimodal distribution, which is an expected result because a number-weighted PDF is much less sensitive to the large grains than an area-weighted PDF.

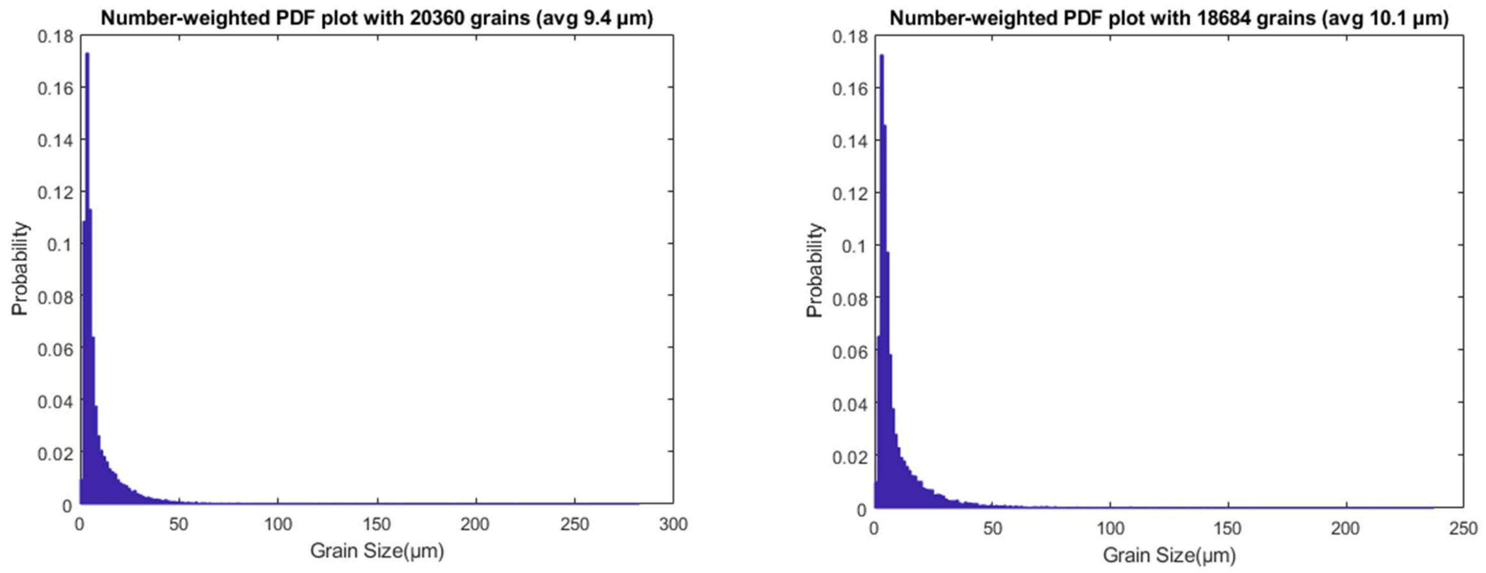


Figure 43: Number-weighted PDF computer for a large region of ceramic sample shown in Figure 38. The left figure is from a local computer while the right figure is from a supercomputer. The grain size is measured via the pixel-count method. Both PDF plots have 200 ranges (X=200).

7. LIMITATIONS

Although the program will work in most cases, it will fail under some circumstances. This section will explore the limitations of our software.

First, the program cannot segment the image when the grain boundaries are not clear enough. For example, it cannot detect the elongated grains on the steel bolt since typically those grains do not have clear grain boundaries. Figure 44 shows an example of elongated grains with unclear grain boundaries.

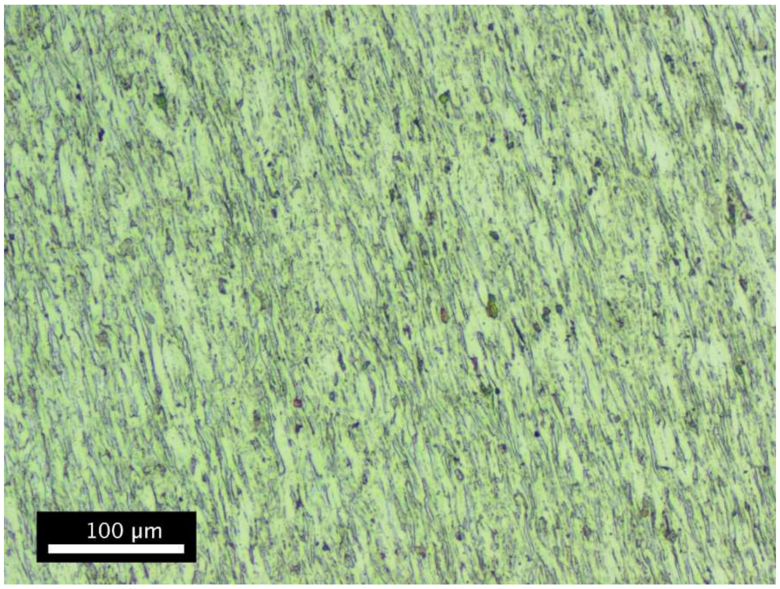


Figure 44: Elongated Grains on the head part of the steel bolt sample imaged with a 20x objective lens.

In addition, there is a tradeoff in grain boundary identification. To eliminate noise, such as the dirt inside the grains, I needed to set a threshold. A big threshold means that it will remove more noise but also increase the possibility of wiping out the actual grain boundaries. A small threshold can help to maintain all the grain boundaries, but it will also retain more noise.

The brightness of the image is another critical factor for grain boundary identification. Due to the light condition on the image, some grain boundaries can be very shallow and hard to detect. Although setting a low threshold in the MATLAB function imbinarize() can help to identify those shallow grain boundaries, it means that a single threshold might work on all the images and hence the program might need some manual work to determine what threshold to use for a specific set of images (e.g., images that contain very shallow grain boundaries).

Another issue is that the same program might detect different grain boundaries on the same spot of the sample from different images (Figure 45). It will lower the accuracy since if the program made one mistake on an image, this mistake might overlay the correct one in the process of stitching.

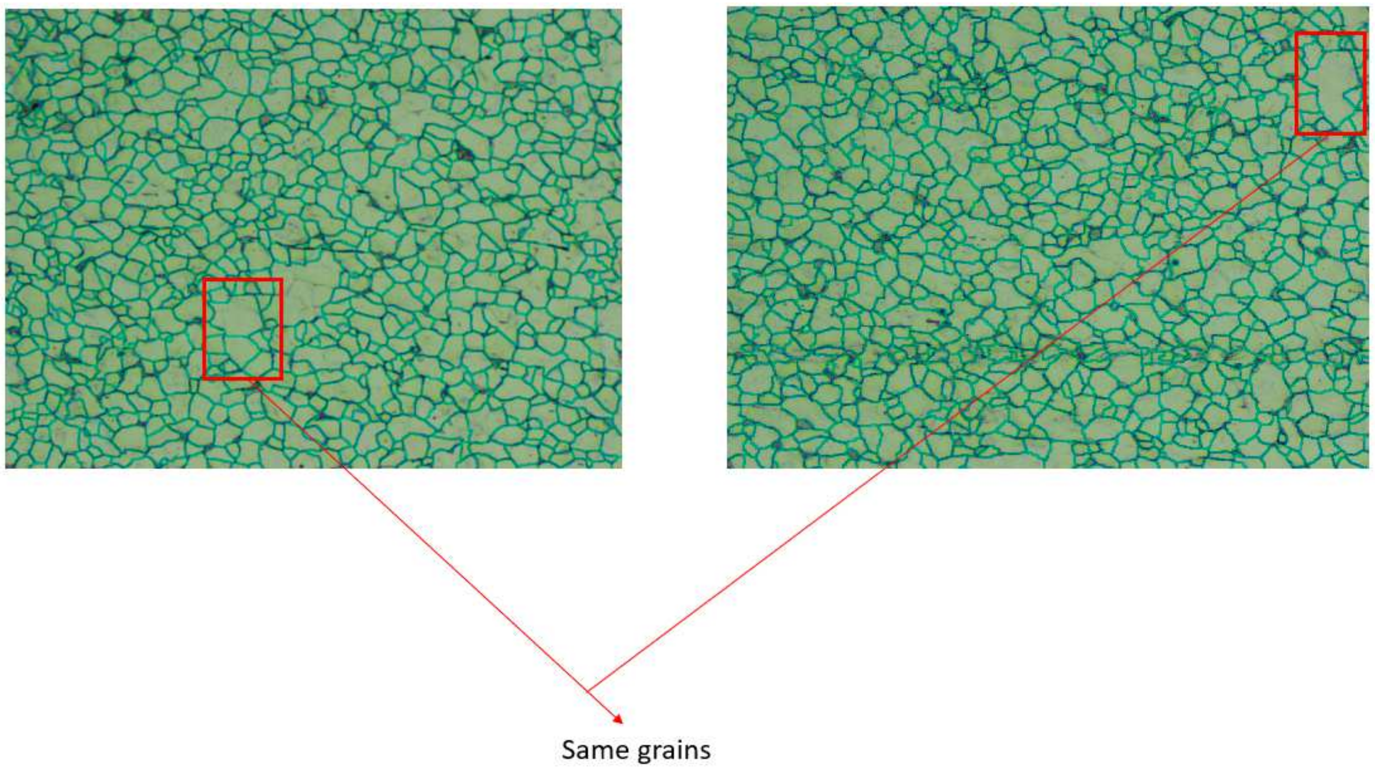


Figure 45: Different grain boundary identification on the same spot

There are several limitations that can lower the accuracy of the grain boundary identification results. First, the dirt on the surface of the sample can affect the performance of the grain boundary identification program. Specifically, my program might consider the dirt as a grain given the similar characterization. Therefore, dirt will increase the number of total grains and divide original large grains into small grains, hence reduce the grain size calculation. Feature 1 in Figure 46 shows an example for the dirt on the ceramic specimen. Second, since the etching process is necessary to make grain boundaries visible to an optical microscope, this process will create Feature 2 in Figure 46, which will be considered as normal grains to the program. Thus, it will cause the problem of affecting the total number of grains and grain size sue to those fake grains. Lastly, the dirt on the microscope objective lenses can also lower the performance of the grain boundary identification program. For instance, Feature 3 in Figure 46 will be considered as edges (grain boundaries) and affect the accuracy of the results. The grain boundary detection image for Figure 46 is shown in Figure 47.

These potential errors become obvious after the grain boundary stitching process. For example, in Figure 48, there are several weird lines, which is caused by the dirt on the microscope's objective lenses, and a minute fake grain, which is caused by the dirt on the sample, in the large grain whose outline is red.

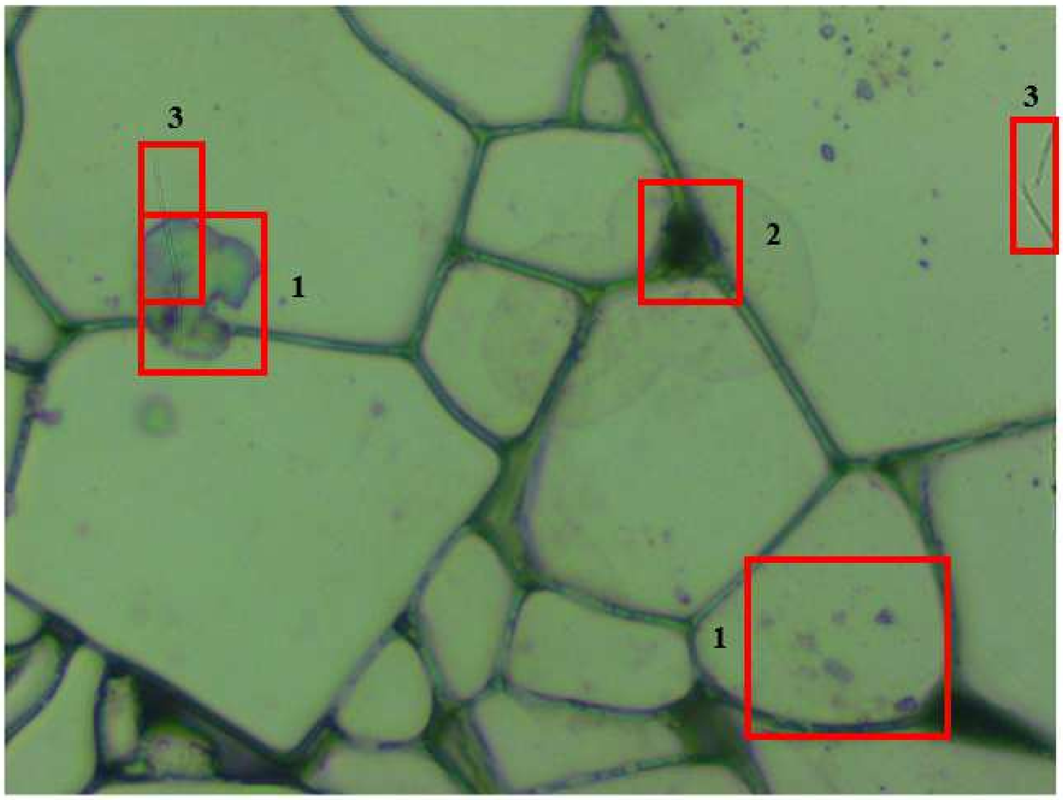


Figure 46: An image from ceramic (100X objective lens). Feature 1 is the dirt on the ceramic sample. Feature 2 appears due to the etching process. Feature 3 is caused by the dirt on the microscope or objective lenses.

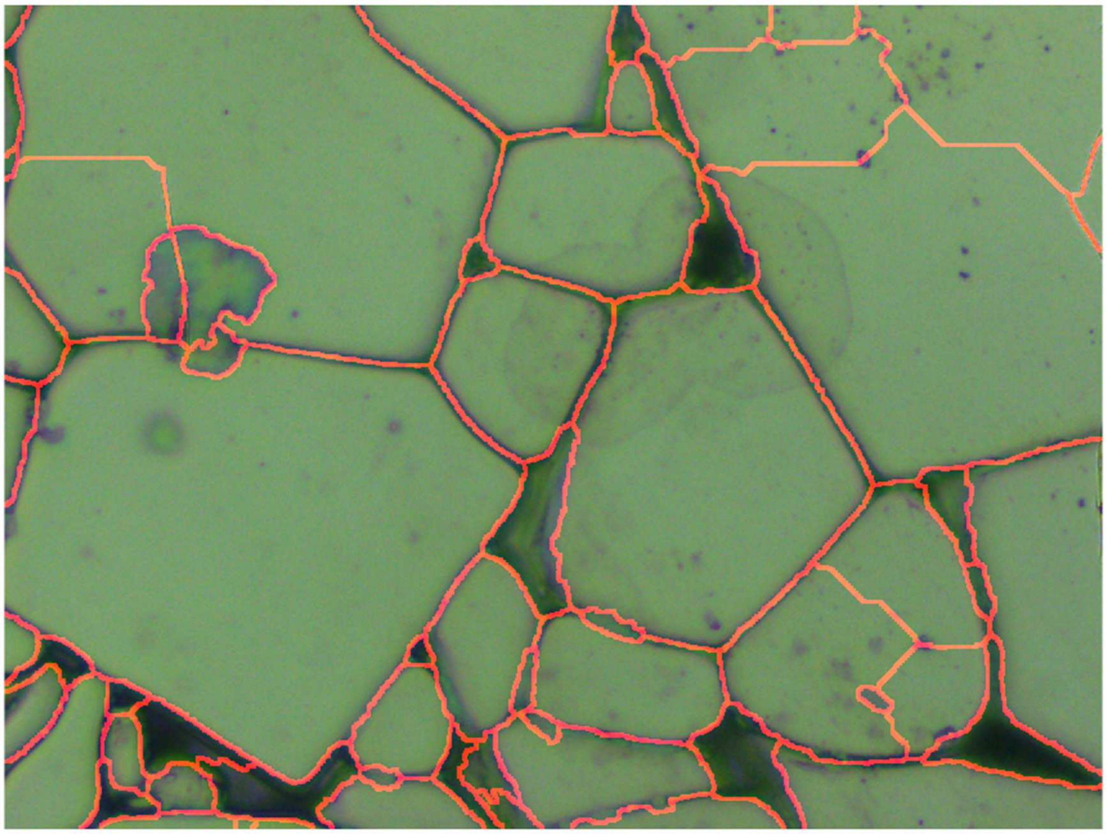


Figure 47: An unsuccessful grain detection result for Figure 46

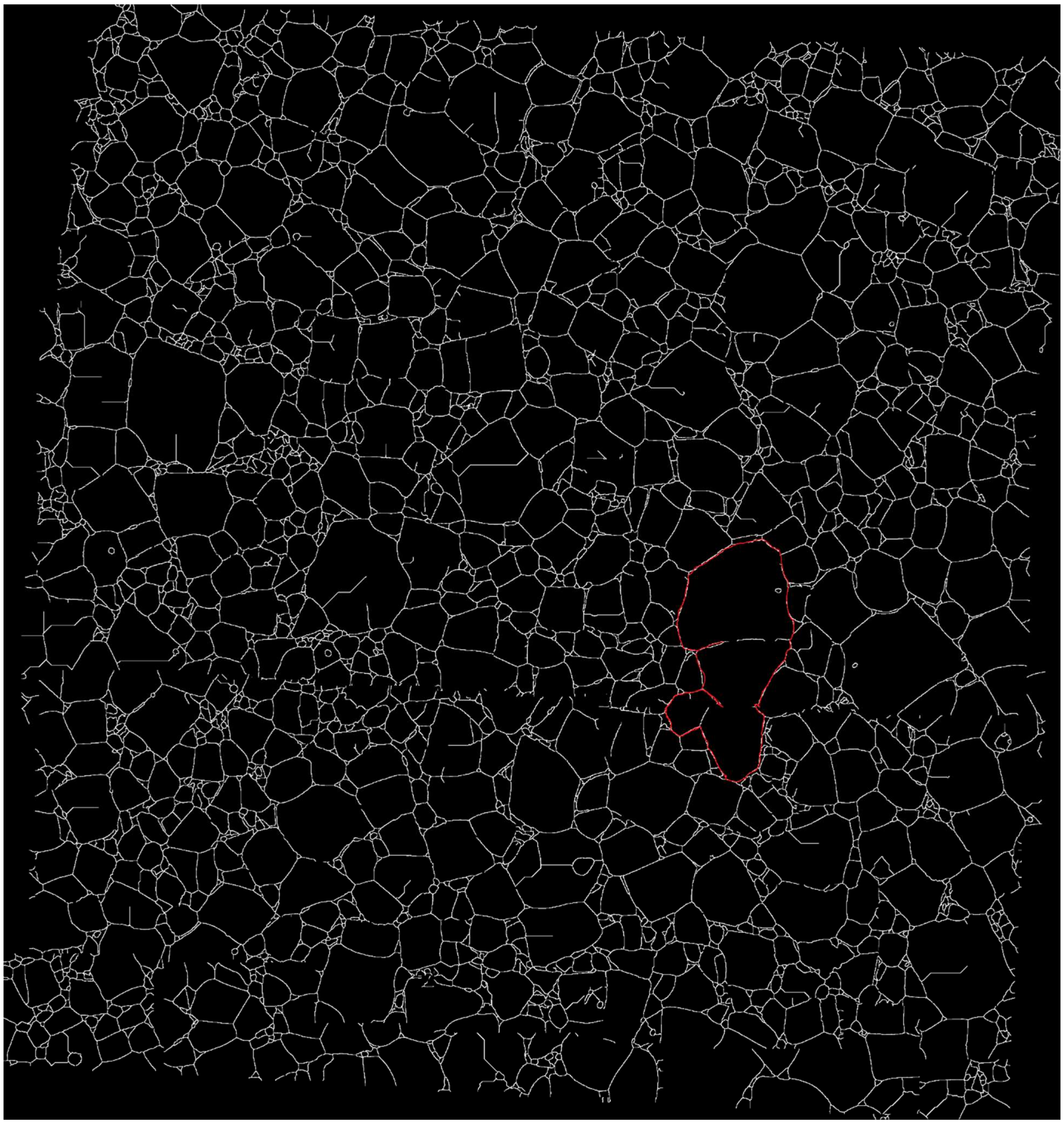


Figure 48: A grain boundary stitched image for a partial region of the ceramic sample. The program will treat the grain with red outline as one grain while it is several grains in fact.

8. CONCLUSIONS

We successfully lowered the cost of measuring all the grain sizes over a large region by making an optical microscope automatically scan a large region of a specimen. Our software not only produces the results that EBSD does, such as view of a large region on the sample and PDF plot, but it also generates the grain size map, aspect ratio map and map of multiple zoom levels. Additionally, our software can also be used to identify “rare events” in microstructure, such as abnormal grains, and flag those rare events in the map for visual inspection.

With our new technology, an optical microscope can measure any kind of material (i.e., either metallic or non-metallic material) automatically with any type of objective lens. Although the grain boundary identification program cannot work for some kinds of images and will make mistakes due to the several reasons described before, it still can return reasonable results, which are precise enough for the grain size analysis and obtaining a distribution of the grain size. Besides, since the PDF results show the expected distribution of grain size on a large area, it suggests that these errors are acceptable.

In addition, the program can be running on both local computer and supercomputer to do analysis on a large stitched image. Although their results are not exact the same, they are close enough so that a user could get a rough idea about the grain size statistics for a very large stitched image from a local computer. Therefore, it solves the problem that when a user does not own a supercomputer, a personal laptop can still be used for automatic analysis work on a large stitched image.

Due to the simple design of the motorized stage, it can be easily modified for another model of microscope. The only things that need to be changed are the size of the motorized stage and the locations and sizes of the holes to install the stage properly. There will be no change for the electrical and software elements.

9. FUTURE WORK

A further comparison between the technique described in this thesis and the existing EBSD technique can be performed to increase the value of this thesis. For example, we can compare the result of mean grain size on the same region from the same sample between each other and compare the cost of doing it. Since both EBSD and SEM are very expensive, we do not have enough funds to purchase them for comparison.

Additionally, a feature that adjusts the lightings on an optical microscope can be added to the current program. This feature can improve the performance of the grain boundary identification by getting rid of shallow grain boundaries.

In addition, better work can be done for the sample preparation because a cleaner sample can yield better results. Further, a cleaner microscope objective lens can also improve the results.

LIST OF REFERENCES

[1] G. B. Rathmayr, A. Hohenwarter, and R. Pippan, “Influence of grain shape and orientation on the mechanical properties of high-pressure torsion deformed nickel,” *Mater. Sci. Eng. A*, vol. 560, pp. 224–231, Jan. 2013.

[2] J. Sosa, P. Sul, and L. Small, “Automated Micrograph Analysis Enables Pioneer R&D,” *ASM Int.*, p. 18, Jan. 2019.

[3] “MagniSci: Our blog on computer vision and how it is applied to microscopy!” [Online]. Available: http://magnisci.com/blog.php. [Accessed: 14-Feb-2019].

[4] P. R. Cantwell, M. Tang, S. J. Dillon, J. Luo, G. S. Rohrer, and M. P. Harmer, “Grain boundary complexions,” *Acta Mater.*, vol. 62, pp. 1–48, Jan. 2014.

[5] N. Shibata, S. J. Pennycook, T. R. Gosnell, G. S. Painter, W. A. Shelton, and P. F. Becher, “Observation of rare-earth segregation in silicon nitride ceramics at subnanometre dimensions,” *Nature*, vol. 428, no. 6984, pp. 730–733, Apr. 2004.

[6] P. Zhang and J. Ruan, “SIFT Algorithm For Image Stitching,” presented at the 3rd International Conference on Material, Mechanical and Manufacturing Engineering (IC3ME 2015), 2015.

[7] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-Up Robust Features (SURF),” *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008.

[8] M. Druckmüller, “PHASE CORRELATION METHOD FOR THE ALIGNMENT OF TOTAL SOLAR ECLIPSE IMAGES,” *Astrophys. J.*, vol. 706, no. 2, pp. 1605–1608, Nov. 2009.

[9] J. Borghoff, L. R. Knudsen, and K. Matusiewicz, “Hill Climbing Algorithms and Trivium,” in *Selected Areas in Cryptography*, 2011, pp. 57–73.

[10] T. Blattner, W. Keyrouz, J. Chalfoun, B. Stivalet, M. Brady, and S. Zhou, “A Hybrid CPU-GPU System for Stitching Large Scale Optical Microscopy Images,” in *2014 43rd International Conference on Parallel Processing*, 2014, pp. 1–9.

[11] J. Chalfoun *et al.*, “MIST: Accurate and Scalable Microscopy Image Stitching Tool with Stage Modeling and Error Minimization,” *Sci. Rep.*, vol. 7, no. 1, p. 4988, Jul. 2017.

[12] N. Otsu, “A Threshold Selection Method from Gray-Level Histograms,” *IEEE Trans. Syst. Man Cybern.*, vol. 9, no. 1, pp. 62–66, Jan. 1979.

[13] N. Dhanachandra, K. Manglem, and Y. J. Chanu, “Image Segmentation Using K -means Clustering Algorithm and Subtractive Clustering Algorithm,” *Procedia Comput. Sci.*, vol. 54, pp. 764–771, Jan. 2015.

[14] S. Beucher, “The watershed transformation applied to image segmentation,” *SCANNING Microsc.-Suppl.*, pp. 299–299, 1992.

[15] F. Meyer, “Topographic distance and watershed lines,” *Signal Process.*, vol. 38, no. 1, pp. 113–125, Jul. 1994.

[16] E04 Committee, “Test Methods for Determining Average Grain Size,” ASTM International.

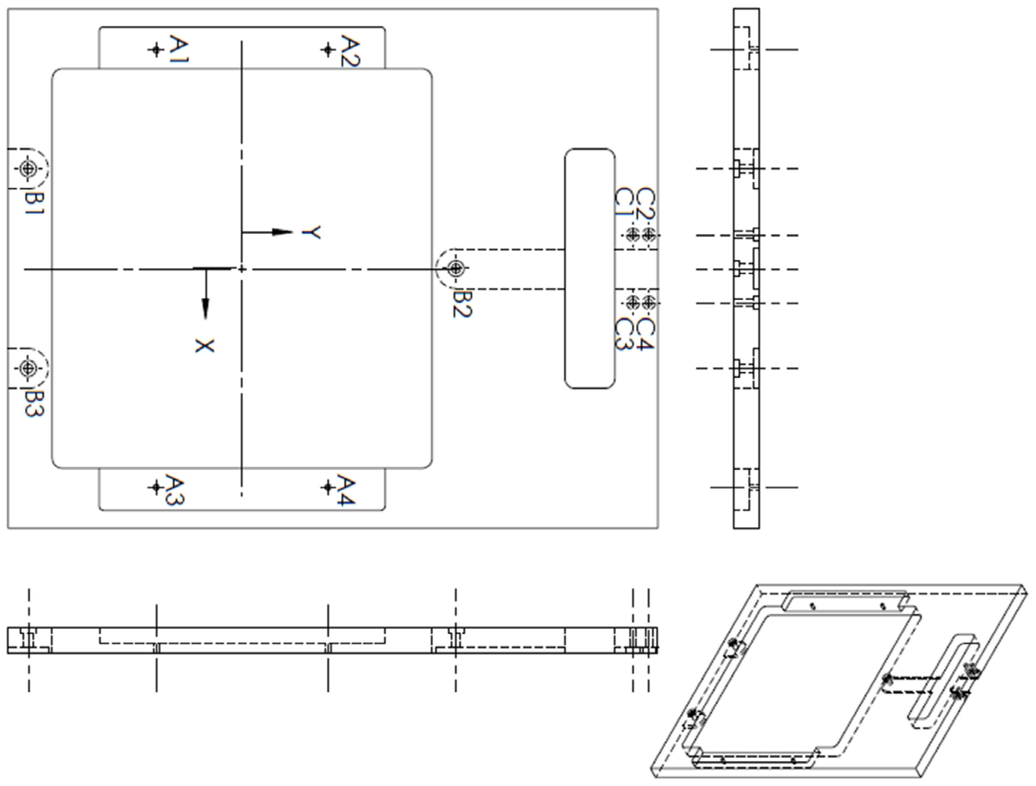
[17] W. D. Callister, J., and D. G. Rethwisch, *Materials Science and Engineering an Introduction 9th edition*.

[18] M. Li, D. Wilkinson, and K. Patchigolla, “Comparison of particle size distributions measured using different techniques,” *Part. Sci. Technol.*, vol. 23, no. 3, pp. 265–284, Jul. 2005.

[19] J. J. Friel, *Practical Guide to Image Analysis*. ASM International (January 15, 2000).

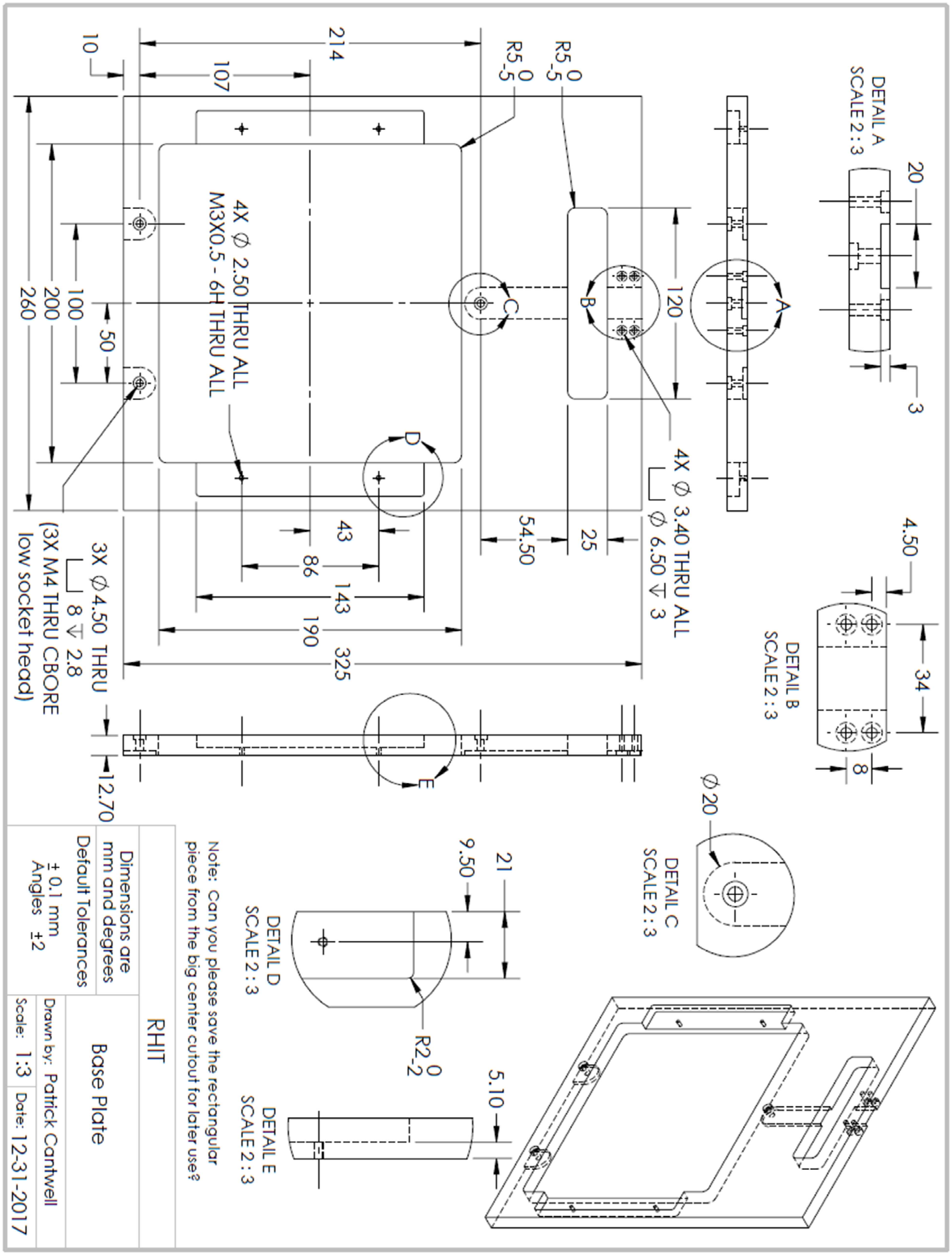
[20] Carl Zeiss MicroImaging GmbH, “Axio Vert.A1 Inverted Microscope Operating Manual,” p. 33.
[21] “MATLAB Support Package for Arduino Hardware - File Exchange - MATLAB Central.” [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/47522. [Accessed: 24-Feb-2019].
[22] “Trapezoidal numerical integration - MATLAB trapz.” [Online]. Available: https://www.mathworks.com/help/matlab/ref/trapz.html#bua4lsr. [Accessed: 02-Mar-2019].
[23] P. Lehto, H. Remes, T. Saukkonen, H. Hänninen, and J. Romanoff, “Influence of grain size distribution on the Hall–Petch relationship of welded structural steel,” *Mater. Sci. Eng. A*, vol. 592, pp. 28–39, Jan. 2014.
[24] P. Lehto, J. Romanoff, H. Remes, and T. Sarikka, “Characterisation of local grain size variation of welded structural steel,” *Weld. World*, vol. 60, no. 4, pp. 673–688, Jul. 2016.
[25] “2D minimal bounding box - File Exchange - MATLAB Central.” [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/31126. [Accessed: 07-Feb-2019].

APPENDIX A



TAG	X LOC	Y LOC	SIZE
A1	-109.50	-43	∅ 2.50 THRU ALL M3X0.5 - 6H THRU ALL
A2	-109.50	43	
A3	109.50	-43	
A4	109.50	43	
B1	-50	-107	∅ 4.50 THRU ⌴ ∅ 8 ↧ 2.80
B2	0	107	
B3	50	-107	
C1	-17	195.50	∅ 3.40 THRU ALL ⌴ ∅ 6.50 ↧ 3
C2	-17	203.50	
C3	17	195.50	
C4	17	203.50	

RHIT	
Dimensions are inches and degrees	Base Plate
Default Tolerances ±0.1 mm Angles ±2	Drawn by: Patrick Cantwell
	Scale: 1:3 Date: 1-1-2018



Note: Can you please save the rectangular piece from the big center cutout for later use?

RHIT

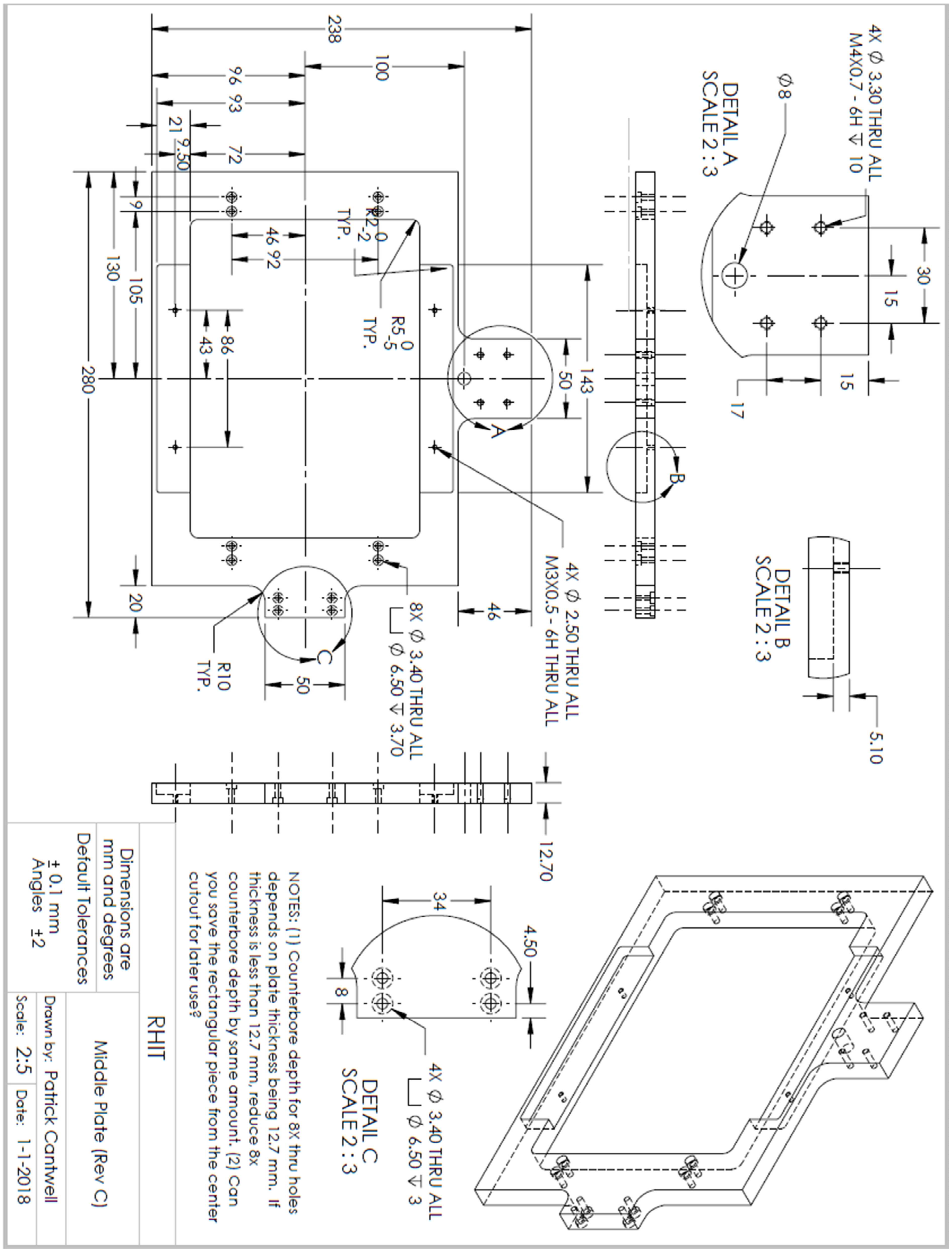
Base Plate

Dimensions are mm and degrees
Default Tolerances
± 0.1 mm
Angles ±2

Drawn by: Patrick Cantwell

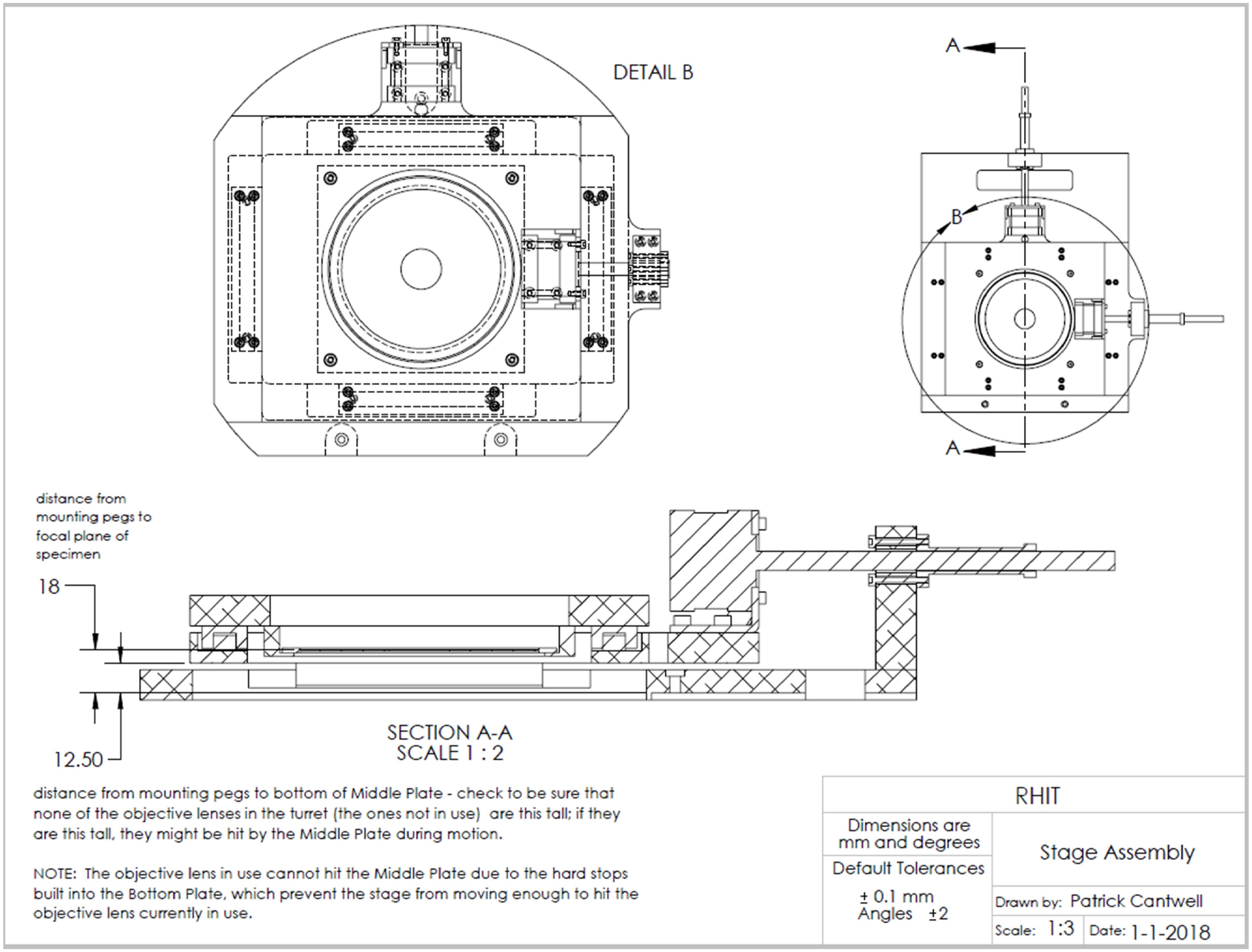
Scale: 1:3 Date: 12-31-2017

APPENDIX B



NOTES: (1) Counterbore depth for 8X thru holes depends on plate thickness being 12.7 mm. If thickness is less than 12.7 mm, reduce 8x counterbore depth by same amount. (2) Can you save the rectangular piece from the center cutout for later use?

Dimensions are mm and degrees
Default Tolerances
± 0.1 mm
Angles ±2

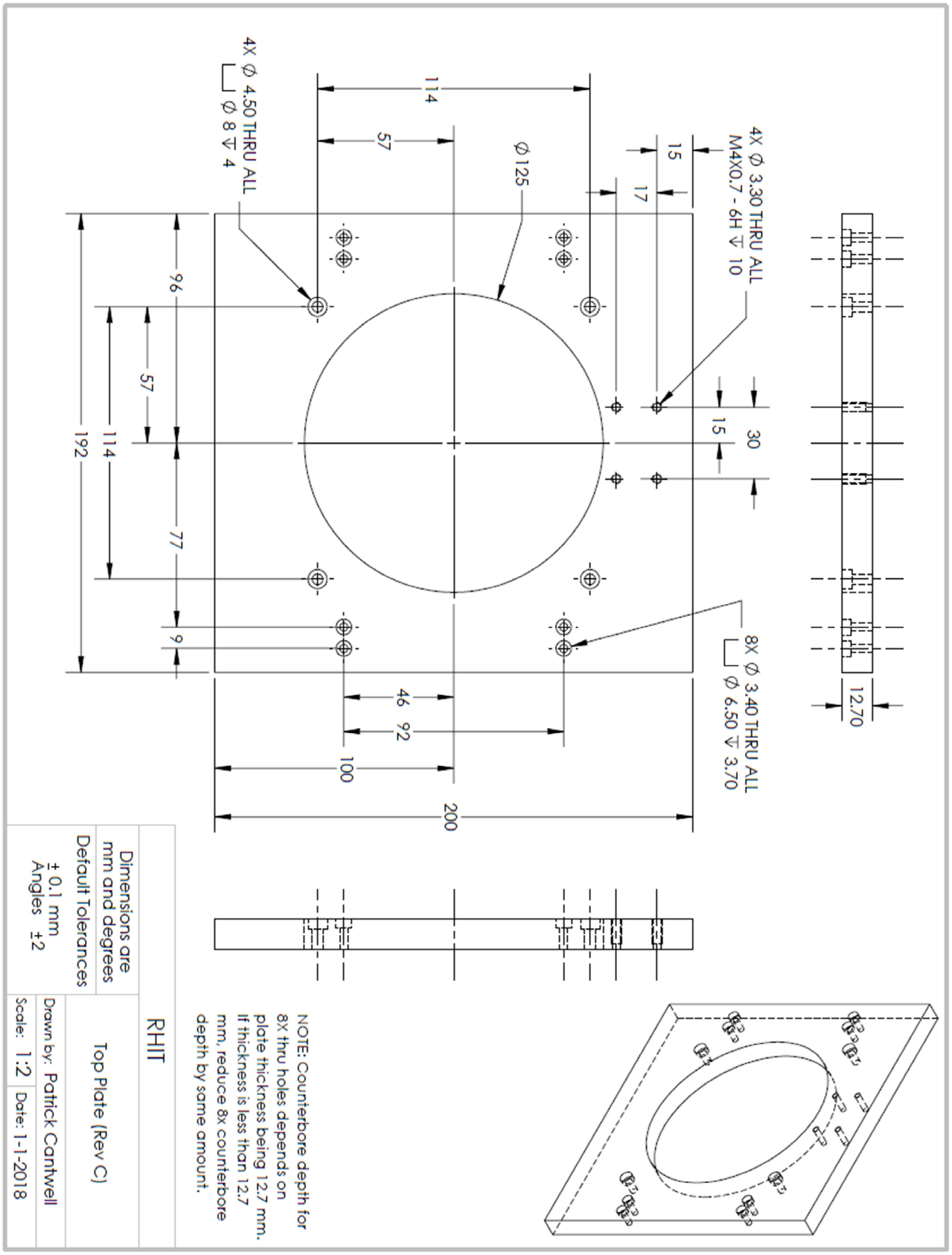


distance from mounting pegs to bottom of Middle Plate - check to be sure that none of the objective lenses in the turret (the ones not in use) are this tall; if they are this tall, they might be hit by the Middle Plate during motion.

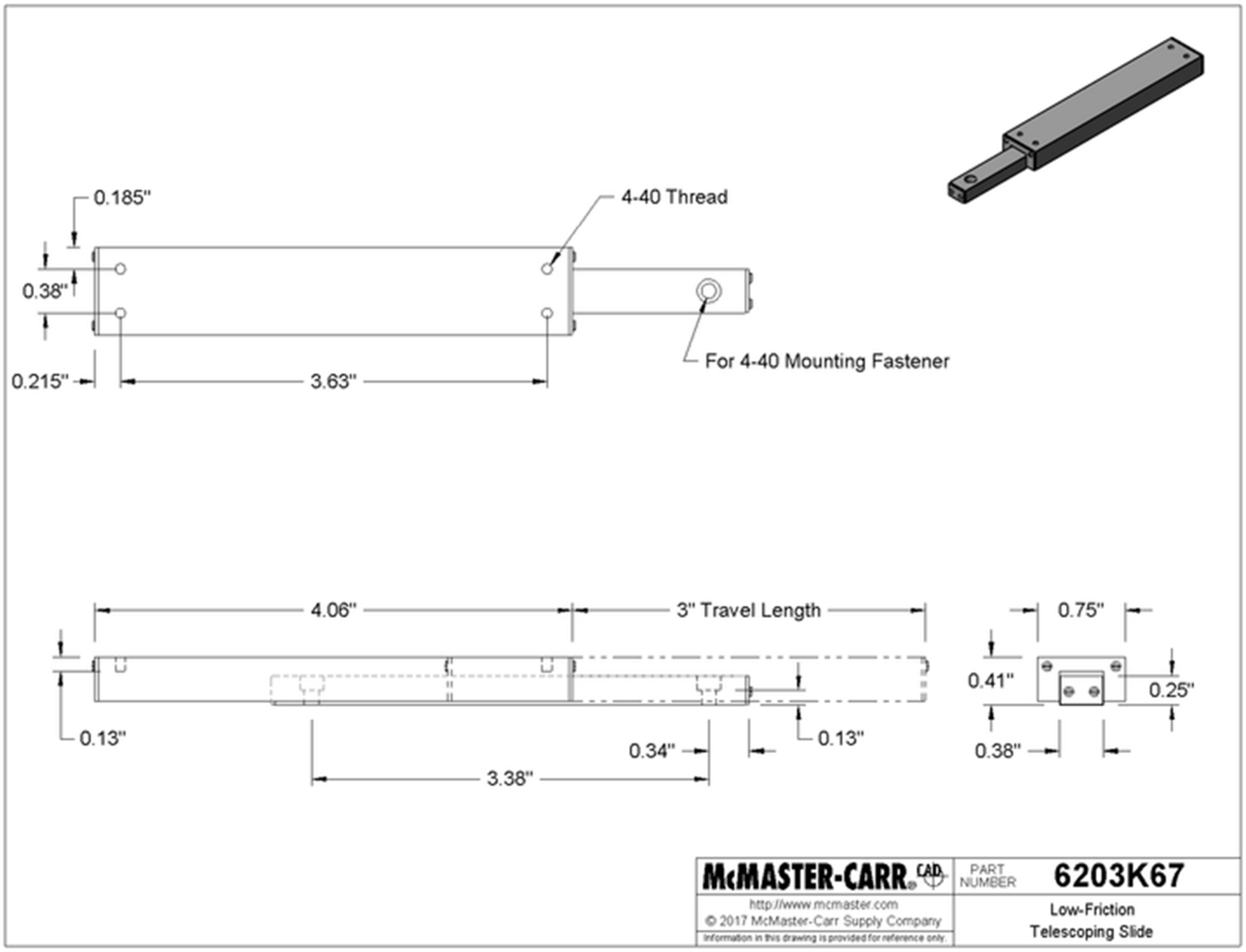
NOTE: The objective lens in use cannot hit the Middle Plate due to the hard stops built into the Bottom Plate, which prevent the stage from moving enough to hit the objective lens currently in use.

RHIT	
Dimensions are mm and degrees	Stage Assembly
Default Tolerances ± 0.1 mm Angles ±2	Drawn by: Patrick Cantwell
	Scale: 1:3 Date: 1-1-2018

APPENDIX C


4X Ø 4.50 THRU ALL
⌴ Ø 8 ↧ 4
114
57
Ø125
4X Ø 3.30 THRU ALL
M4X0.7 - 6H ↧ 10
15
17
96
57
192
114
77
9
15
30
8X Ø 3.40 THRU ALL
⌴ Ø 6.50 ↧ 3.70
12.70
46
92
100
200
NOTE: Counterbore depth for 8X thru holes depends on plate thickness being 12.7 mm. If thickness is less than 12.7 mm, reduce 8x counterbore depth by same amount.
Dimensions are mm and degrees
Default Tolerances ± 0.1 mm Angles ±2
RHIT
Top Plate (Rev C)
Drawn by: Patrick Cantwell
Scale: 1:2
Date: 1-1-2018

APPENDIX D


0.185"
4-40 Thread
0.38"
0.215"
3.63"
For 4-40 Mounting Fastener
4.06"
3" Travel Length
0.13"
0.34"
0.13"
3.38"
0.75"
0.41"
0.25"
0.38"
McMASTER-CARR
http://www.mcmaster.com
© 2017 McMaster-Carr Supply Company
Information in this drawing is provided for reference only.
PART NUMBER
6203K67
Low-Friction Telescoping Slide

The linear bearings are Low-Friction Telescoping Slide 6203K67 from McMaster-Carr.