Automated Fiber Measurement System

ME 470 Spring 2017 - Final Report

University of Illinois at Urbana-Champaign

Henry Cuzco

Sam Halvorsen

Da Woon Kim

Alan Luo

Yi Jie Tan

Project Sponsor: Monahan Filaments

Mr. Jon Monahan

Submitted May 5, 2017

Table of Contents

I. Executive Summary

The purpose of our project is to design a fiber measurement system using image processing techniques to measure out 20 fibers in under two minutes.

The fixture we constructed consists of a 14 megapixel camera, a secondary plano-convex lens, multiple fiber trays, and an enclosure for light isolation. For lighting, an LED ring is used to provide even lighting across all the fibers, removing shadows for clear images. The algorithm, written in MATLAB, is the foundation of the project and measures the dimensions of multiple fibers by binarizing the image and converting the pixel resolution to physical distances. To make this program easy to use, the code was structured into a user-friendly graphical user interface (GUI). The GUI has been tested and works on computers running the Windows 7, 8, and 10 operating systems, including those without a MATLAB license. There is also a manual in the form of a readme file which contains instructions for installing all the necessary software and an explanation of the GUI and AmScope software.

The construction of the fixture was done through experimentation in a lab setting. The parts needed for the fixture were purchased as needed, while the tray designs were continuously modified to allow for optimal positioning of the fiber strands. Similarly, the code had to be redeveloped to ensure that the code is able to distinguish between the fiber strands and the shadows that resulted from the lighting.

Most of the expenses for this project came from purchasing optomechanical parts for the fixture, such as buying the camera, posts, and enclosure. Other expenses include travel to the Monahan factory. The total expenses came out to $920.38, staying under the allotted $1500 budget.

While we were successfully able to meet the project and budget quotas, there is future work that can be done to improve the performance of the measurement system. A camera with a standard focal length and auto-focusing capabilities would allow us to decrease the height of the fixture, conserving both space and money. Modifying the GUI so that it is able to run optimally on the sponsor's computers is also a necessary step. Finding a way to automate the process of taking a picture and processing it all at once would streamline the entire process of the measurement system. Finally, further adjustments to the code can be made if requested.

II. Introduction

Monahan Filaments manufactures a wide range of plastic monofilaments for industrial purposes. Their primary factory includes several production lines for fibers varying in color, size, and material. Some applications include cosmetic brushes, tooth brushes, automotive aftercare, as well as cement strengthening fibers. Many of their fibers are synthetic and crimped, with diameters ranging from 0.001–0.080 inches. Figure 1 displays some of the many types of fibers produced at Monahan.

Figure 1. Multiple fiber sizes, shapes, and colors

Fibers are quality checked for consistency in dimensions of thickness, crimping amplitude and frequency. Current manual measurement methods of quality control are often long and tedious, taking up to 20 minutes per batch of 20 fibers. Thickness of fibers are measured using a digital micrometer. The fiber's shadow is projected below onto a ruler, and the amplitude is measured directly from the ruler and the frequency is calculated using the calculator. Monahan Filaments seeks for an automated solution to reduce the quality check time to approximately 2 minutes per batch of fiber through a fast image processing algorithm.

Our team proposed to use a high resolution camera held in a fixture to capture an image of several fibers, and subsequently run them through a MATLAB algorithm to determine the thickness, crimping amplitude and crimps per inch (CPI). The primary objective of this project was to deliver a working prototype of a system that can drastically reduce the measuring time of fibers from 20 minutes to 2 minutes. This will save Monahan several hours The project was divided into two main teams: one to design and manufacture a mechanical fixture for the measuring station and another to develop a graphical user interface with simple controls that utilize the fast MATLAB algorithm. Since several aspects of this project were open-ended, multiple possible solutions were devised.

III. Proposed Solutions

A. Optics

The camera is one of the most important facets of the entire fixture. Without a high quality camera to capture the fibers, accurate image processing and measurements would not be possible. The main requirements of our camera was that it would have high resolution, easy connection to a computer (via USB), and MATLAB integration. We performed initial testing with phone cameras aiming to figure a minimum resolution for the camera we would ultimately purchase. We discerned that an iPhone camera with a 12 megapixel (MP) resolution is capable of producing high quality images of the sample fibers given to us by Monahan Filaments. Since the camera requires a high resolution, we ruled out typical commercial webcams, which rarely have resolutions over 2 MP.

Through our research, we found and purchased an AmScope MU1403 scientific-grade camera. This camera has a 14 MP resolution and USB 3.0 connection. However, it cannot be used natively in MATLAB. Luckily, the AmScope camera software is easy to use. It displays a live feed of the camera along with a multitude of settings including exposure, white balance, brightness, and contrast. For this project specifically, we believe that most of these settings apart from the exposure can be set to default. We typically set it to “auto-exposure” allowing the camera to choose the optimal setting based on the lighting in its environment. We also change the image to be black and white from color for ease of viewing. Additionally, since this camera is

typically used for microscopy, its focal length is extremely small. For this reason, a secondary lens placed in front of the camera is necessary to provide a reasonable focal length.

Our advisor Dr. Kimani Toussaint allowed us to experiment with the fixture design in his lab, including using different lenses to increase the focal length of the camera. The main two lenses we tested were 52 mm and 38.1 mm plano-convex lenses, both one inch in diameter. We settled on the 38.1 mm lens because its pros outweigh its cons. Though the 38.1 mm lens requires a taller fixture, it provides a better image than the 52 mm lens, which exhibits a more noticeable fisheye effect. This is likely due to its geometry, as it is much more concave than the 38.1 mm lens. The fisheye lens distorts the image and by association, the fibers themselves by warping their shape and blurring the edges. The 38.1 mm lens exhibited this deficiency as well, but to a lesser degree. To allow for slight adjustments, we placed the lens inside an adjustable tube purchased from Thorlabs. In order to achieve a reasonable field-of-view, the camera is located approximately 15/16” from the top of the lens, and the bottom of the lens tube is located 18.75” above the fiber tray. This produces a field-of-view with dimensions 4” by 5”. Figure 2 below contains both the camera and the lens set up on the base of the fixture.

Figure 2. Camera and lens held up above fixture base

Figure 3 contains an example picture taken using the above setup (along with proper lighting and enclosure to be discussed later). This image is high quality, save for the fibers on either end. Some blurring and warping occurs at both the left and right ends, which is likely due to slight imperfections in lining up the camera and lens with the fiber tray.

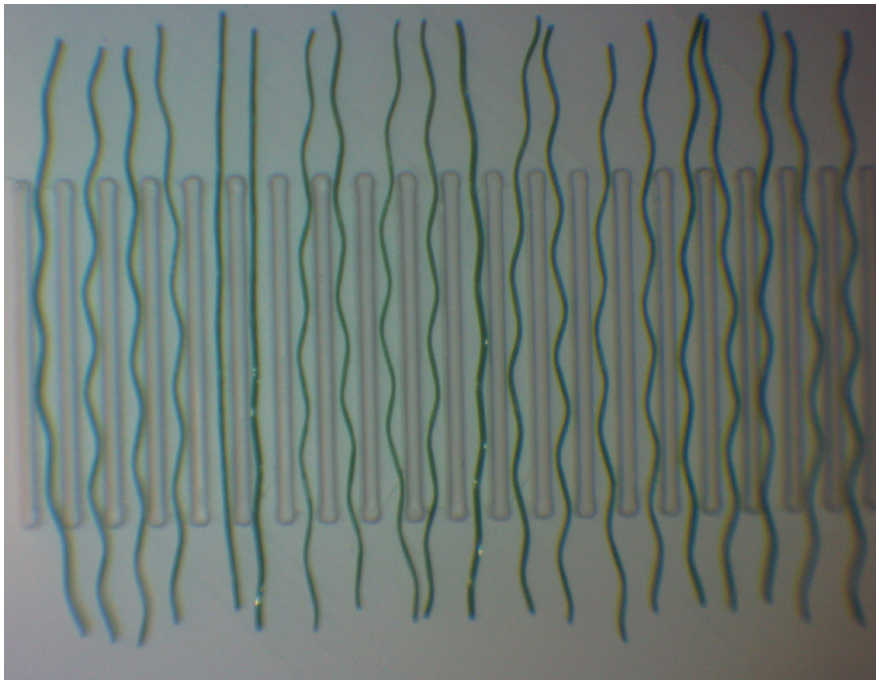

Figure 3. Example image of green fibers. 0.028 diameter. Note slight warping on left and right.

B. Lighting

In addition to the camera, the lighting system of this system is key to producing high quality images. To accurately measure the fibers through image processing, they cannot have visible shadows from above, nor can there be substantial reflections on the fibers. We chose an LED ring to act as our lighting source. The power input of the LED ring is 110-240 V, 50-60 Hz with auto-switching. When placed directly under the lens tube, it provides even lighting over the fibers. However, even setting it at a very low brightness setting using its adjustability knob resulted in too much light reflection off the fibers. We purchased spray paint to mimic "frosted glass" and used that to help diffuse the light to reduce reflections. Figure 4 below shows the original LED ring and the modified version with the frosted glass finish.

Figure 4. LED ring and its modification.

One alternative lighting system (and by extension the tray design) considered was to instead place the light source below the fiber tray. A different design for a translucent tray would then be created to both allow light through itself and remove the possibility of light reflections from the fibers. We designed a tray with short legs which would leave space for LED strips to lay flat below. This tray was supposed to be printed in the Ford Laboratory in the Mechanical Engineering Laboratory using stereolithography. However due to setbacks in the Ford Lab, the tray has yet to be printed. For this reason we abandoned this lighting method and continued to use the LED ring placed above the tray.

C. Enclosure

After deciding on the optics and lighting method, we designed an enclosure that would hold the different components. We used the length of the fibers as a minimum constraint for the width of the fixture and the distance from the camera and lens to the fibers for determining the height. Once we decided on the dimensions, we purchased optomechanical parts from Thorlabs to ensure stability and quality, while providing easy adjustments that were crucial in our testing period. With an aluminum baseboard of dimensions 6”x12”x1/2”, we fixed a 24” vertical pole, on which we attached the camera level with the top and the adjustable lens holder 15/16” below.

We made sure that the camera and lens were properly aligned with each other to capture the fiber tray below. To block out ambient light and guarantee consistency between tests, we used black hardboard to create walls and a ceiling, held up by structural rails. Slots were cut out of these walls to hold a platform for the LED ring to be placed on. Figure 5 below shows a photo of the final enclosure.

Figure 5. Interior view of fixture, one wall removed.

Initial designs of the enclosure consisted of building a frame and platforms from wood or custom machined parts. However, these were discarded early on due to the difficulties of working with wood. Proper alignment was too important a factor to allow tolerance issues that arise with using wood as a building material. In addition, custom parts would likely be too expensive for our budget of $1500. After using the optomechanical parts available in Dr. Toussaint's lab to test our design, we decided that they would work well in the final fixture.

D. Tray

The tray design underwent several iterations to optimize fiber placement. Since the field-of-view of the camera in its current state is about 4” x 5” and the tray needs to hold 20 fibers, they must be very close together. All of the design iterations involve small walls on a flat base to separate the fibers and guide the user into placing them neatly. In addition, they are spray painted either matte black or matte white to maximize contrast relative to the fibers. The first three major iterations below in Figure 6 reflect on the overall tray design process. The initial design is only 1.5” wide, causing fibers to bend downward on either side. The subsequent designs decreased wall size from 5” to 3.81” and widened the base. This saves space by putting the fibers close together, and also allows them to lay flat.

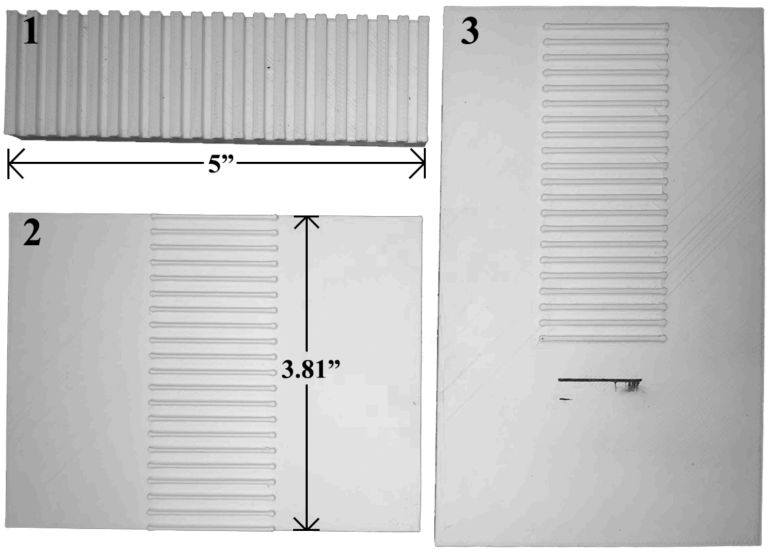

Figure 6. Previous tray iterations. Size increased to allow fibers to lay flat and for tray to fit perfectly within the fixture.

The final tray design in Figure 7 shows two variants, one black and one white. The most prominent change in this design is the wall relocation from the center of the tray to the edge. Testing done with previous designs led to issues in the image processing. The fiber detection algorithm sees the walls as objects, and measures them in addition to the fibers. Thus the walls were moved, and the fiber detection was modified to only detect fibers in the region outside of the walls (to the left or right depending on the tray). Another important feature added to the final design is the scale bar. Each tray has a 1” scale bar located below the fiber placement region.

These have two functions. The first is to convert pixels to inches when the image is run through the algorithm. Additionally, it aids in the fixture calibration process. If the bar is blurry or does not look flat, the fixture must be further calibrated to ensure the highest quality image is obtained.

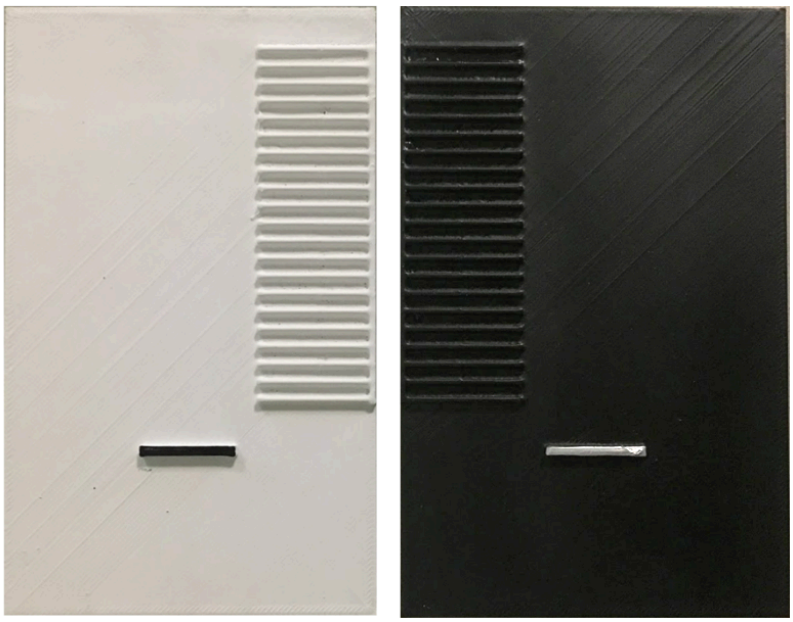

Figure 7. Final Tray designs. Note that walls are placed on the side, the scale bar (1" long) placed below, and the matte finish.

Multiple alternate tray designs were also considered in addition to the translucent concept discussed in the lighting section. In an attempt to remove the walls from the tray during image capture, multipart designs with removable walls were created using CREO software. One such design shown below in Figure 8 contains two parts: a flat base and a comb-like structure which can be removed after the fibers are placed.

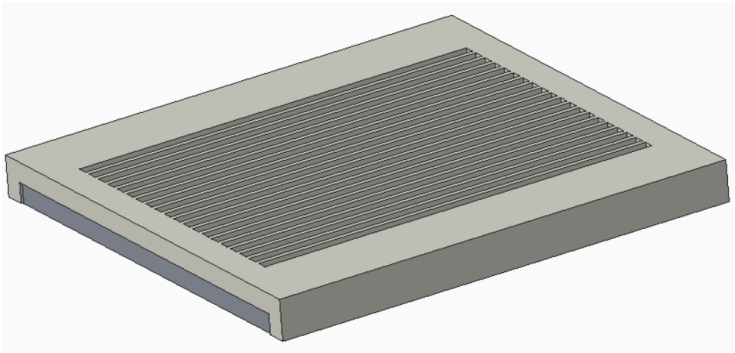

Figure 8. Comb design. The light-gray part is lifted after fibers are placed leaving a flat surface with aligned fibers.

The other design include walls that when placed in the fixture, drop down such that they lay flush with the base containing the fibers. Figure 9 contains this design, showing the walls both elevated and lowered on the left and right respectively. These trays were never printed as we instead opted to keep the design as simple as possible, with less chances for the fibers to move around and therefore disturb the image capturing process.

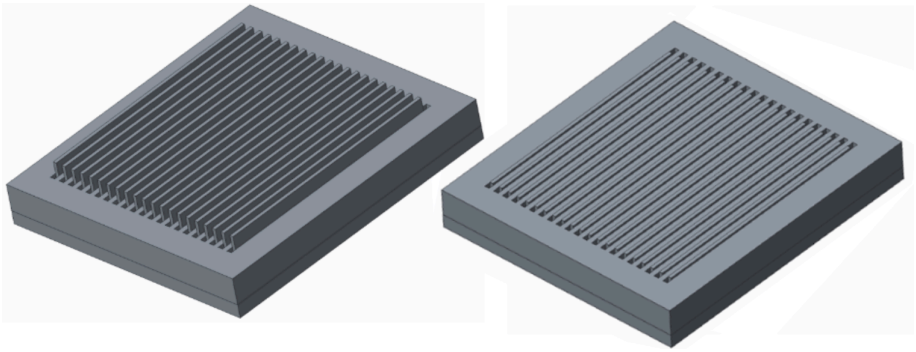

Figure 9. "Fall-through" design with the walls elevated (left) and the walls flush (right).

E. Image Processing Algorithm

The image processing algorithm that was developed allows the user to measure up to twenty fibers simultaneously given an image of fibers. The algorithm is able to automatically measure the thickness, amplitude and the crimps per inch (CPI) of the fibers. The program runs extremely fast and is able to process a picture of multiple fibers under one second. In addition, a user-friendly graphical user interface (GUI) was developed so that the user can easily work through the program. Step-by-step instructions are provided on the GUI, from browsing and loading the fiber image, to entering the nominal values for key measurement data of the fibers.

The GUI is able to display the measurement data of each individual fiber in a table format, with the mean and standard deviation of the measurement data displayed as well. Also, the program is able to identify fibers whose measurements lie outside of the tolerances of the input nominal values. The fibers that lie outside of the tolerances are tagged on the original image, which is displayed on the GUI. In addition, all of the measurement data are exported in Microsoft Excel spreadsheet files so that they can be referred to at a later time.

In the image processing algorithm, there are multiple steps involved in order to obtain the measurement data of the fibers. The major steps involved are:

1. Thresholding of the image
2. Separating pictures of individual fibers
3. Obtaining the one-dimensional waveform that describes the shape of each fiber
4. Performing a Fourier transform to obtain amplitude and spatial frequency in terms of pixels
5. Converting pixel length to physical dimensions.

The first major step is to threshold the image, which is the process of partitioning an image into its foreground and background components. To do so, the original colored images of the fiber, with an example shown in Figure 10, have to be converted to grayscale. This is so that each pixel will have its own associated intensity value, depending on its color in the original image.

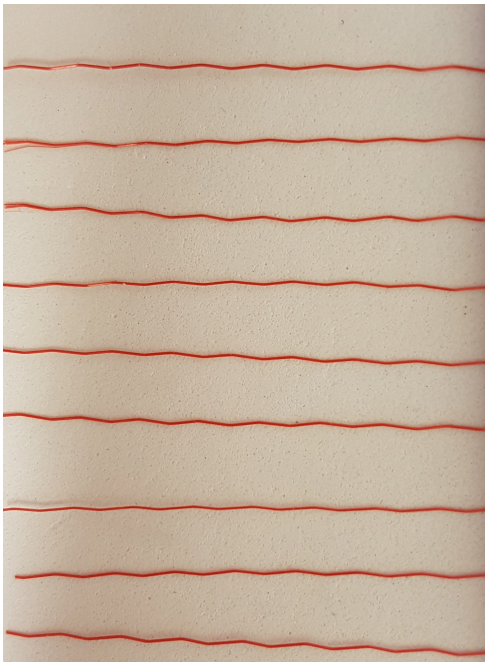

Figure 10. Example input image, before processing.

Thereafter, the grayscale image is then converted into a pure black and white image (also known as binary image), as shown in Figure 11. This process is known as image binarization,

and turns foreground elements into white pixels and background elements into black pixels. In this case, the foreground elements are the fibers, and the background is the loading tray for the fibers. For clear fibers on a dark background, the binarized image has to be inverted so that foreground remains as white and the background remains as black.

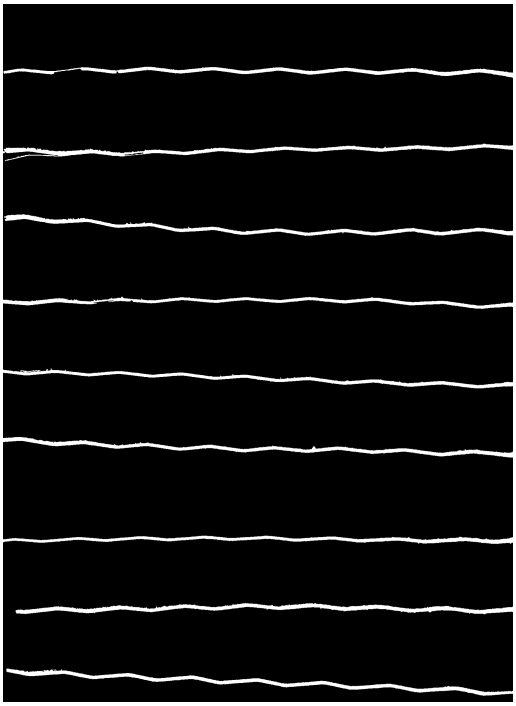

Figure 11. Binary Image, consisting only of black and white pixels.

The binarization method used in this algorithm is the local adaptive method, which determines a threshold value for every pixel. The *imbinarize* function from the Image Processing Toolbox in MATLAB is used in this algorithm. If an intensity value lies above this threshold value, it is converted to white, and vice versa to a black pixel. This threshold value is determined by using the local mean intensity around the neighborhood of every pixel. The size of the neighborhood is defined to be one-eighth of the size of the image.

The next major step is to split up the binarized image of multiple fibers into separate images of individual fibers. This is done by obtaining the smallest rectangular box that contains each fiber, using the *regionprops* function in MATLAB. Using the position of the boxes, the fibers can be cropped out from the image. In addition, the angle that each fiber makes with respect to the horizontal axis is obtained, also with the *regionprops* function, so that they can be oriented parallel to the horizontal axis. The rationale for rotating the fibers to a horizontal

position is so that the one-dimensional waveform of the fiber can be easily obtained. This waveform is expressed in terms of the pixel vertical position on the y-axis as a function of its horizontal position on the x-axis. To obtain this, the top and bottom boundaries of the binarized fiber image were extracted. Then, a waveform that is centered in the fiber is obtained by taking the midpoint vertical position between the top and bottom boundaries throughout the length of the fiber. The result of this is shown in Figure 12.

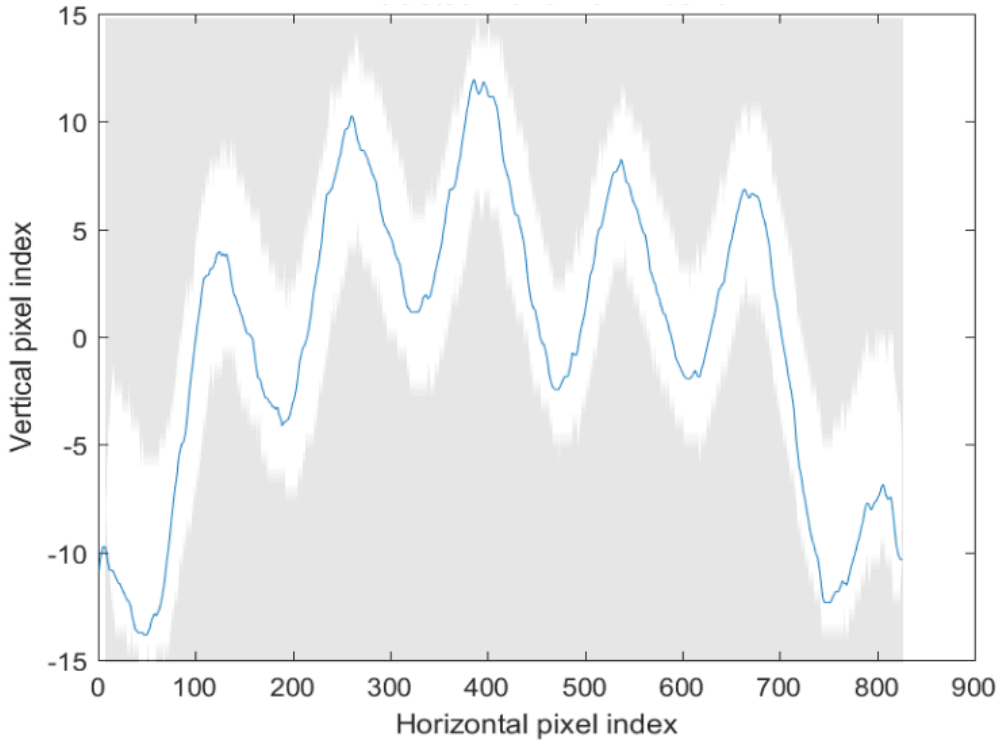

Figure 12. One-dimensional waveform. Constructed by taking midpoint values throughout the length of the fiber.

In addition to obtaining the fiber waveform, the thickness of the fiber can also be measured. This is done by taking the difference in vertical pixel positions of the top and bottom boundaries throughout the length of the fiber. This gives an array of values with the same number of elements as there are horizontal pixels. The average of these values is computed and is taken to be a measurement of the thickness of the fiber, in terms of pixels.

After obtaining the centered waveform of the fiber, we can then take its Fourier transform in order to extract spatial frequency and amplitude of the fiber. The Fourier transform segments the waveform signal into a linear combination of basis sinusoidal functions, each with its own

associated amplitude and frequency. In the program, the function *fft*, or fast Fourier Transform algorithm, was used to compute Fourier transform of the waveform. With this information, we are able to plot a magnitude spectrum of the fiber waveform in terms of pixels as a function of frequency, with units of cycles per pixel. An example of a magnitude spectrum of a fiber is shown in Figure 13.

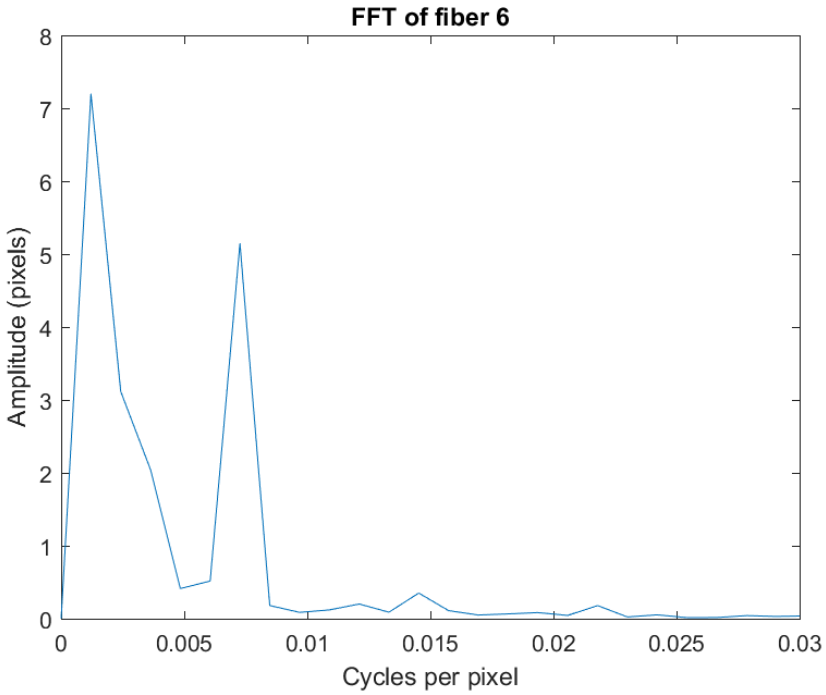

Figure 13. Magnitude spectrum of an example fiber.

From the graph, it can be observed that there are two peaks in the plot. This indicates that there are two frequencies with predominant amplitudes present in the fiber waveform. The peak corresponding to the lower frequency is attributed to any bending or curvature of the entire fiber sample, which is ignored in this case. We are however interested in the second peak with a higher frequency component, which will be able to provide us with the crimping behavior of the fiber. Hence, the amplitude in pixels and the frequency in cycles per pixel is then extracted from the position of the second peak.

With the values of thickness, amplitude and frequency obtained in terms of pixels and cycles per pixels, they will then be converted to inches and CPI. For both thickness and amplitude, the conversion is straightforward and is done by multiplying a scaling factor. The scaling factor from pixels to inches is obtained from the length of the scale bar on the fiber tray

in pixels and taking its inverse. For the spatial frequency, its value has to be divided by the same scaling factor to obtain cycles per inch. However, since there are two crimps per full cycle of the waveform, this value is multiplied by two to obtain CPI.

The MATLAB code for the main body of the image processing algorithm can be found in the appendix.

The accuracy of the image processing algorithm was determined by comparing the data obtained from the program and comparing it against data obtained by measuring the fibers physically. The physical measurements were taken with a vernier caliper with a precision of 0.0005”. Nine fibers were used for testing, and the results are summarized in Table 2 in the appendix.

While majority of the error percentages of the measurements are under 10%, there are some instances where the error percentages for the amplitudes and thickness exceed that value. These errors can be attributed to the presence of reflections and shadows that show up on the image of the fibers on the fiber tray. Reflections on a fiber can cause the thresholding process to mistakenly binarize it as a white region. Shadows that appear on around the fiber can cause the pixels to be binarized as black, thus incorrectly increasing the value for thickness. In addition, the physical measurements are also very prone to human error since the dimensions that are measured are extremely small.

An initial alternative to the image processing was using a Fourier Transform in 2 dimensions. Like the 1 dimensional Fourier Transform, the 2D transform decomposes an image into its sine and cosine basis images. In 2 dimensions, we can relate a sine image to patterned stripes. The difference here compared to the 1 dimensional case is that the frequency is defined both horizontally and vertically. Taking the Fourier transform of this image, the resulting image becomes a series of points where the position relative to the center determines the horizontal and vertical spatial frequencies. The magnitude associated with the horizontal and vertical frequency is represented with the intensity value of the point. The main reason that this approach was discarded was that conducting a 2D Fourier transform on an image of a one-dimensional sinusoidal curve does not return the value of the magnitude and frequency of the one-dimensional component function.

F. User Interface

The 1-D spatial frequency image processing algorithm is incorporated into a simple to use graphical user interface (GUI) with step by step instructions written on the interface of the program. A screenshot of the GUI is shown below in Figure 14.

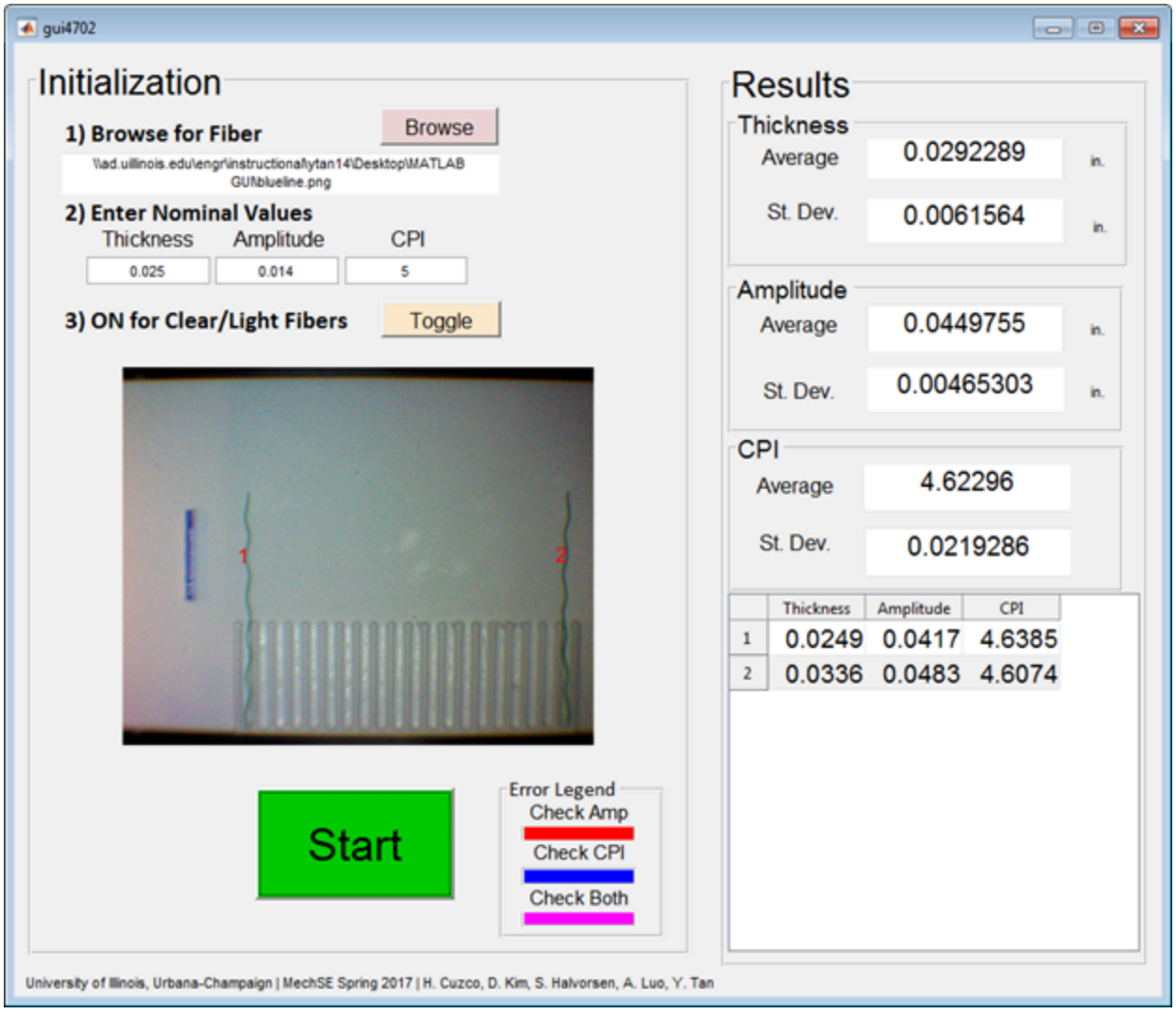

Figure 14. Graphical User Interface.

The first step in using the GUI is to browse for a file either in .jpg or .png format. If a file is not entered, an error message will be generated prompting the user to search for a file, as shown in Figure 15.

Figure 15. Close up of the browsing panel with error message.

Next, the nominal values must be entered. This is done so the program can identify any fibers that deviate from the tolerance of the nominal value standards. If the nominal values are not entered, then an error message will be generated prompting the user to enter the values, as shown in Figure 16.

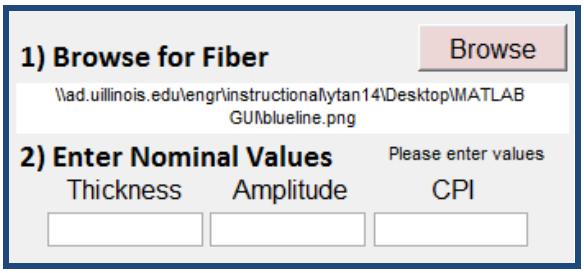

Figure 16. Close up of the nominal values panel with error message.

The third step is to toggle on if clear or light color fibers are being measured. For dark and opaque fibers, this step may be ignored. Once toggled, the button would turn grey so the user can identify which mode the program is running, as shown in Figure 17**.**

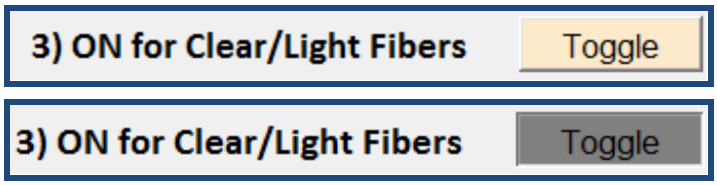

Figure 17. Close up of the nominal values panel with error message.

Finally, after pressing start, the results will be displayed on the right half of the screen. The average and standard deviation of the thickness, amplitude, and crimps per inch will be shown. A table displaying the measurements of each individual fiber is shown below. Furthermore, an excel spreadsheet will be automatically generated for each fiber data measurement.

The numbers that correspond to each fiber measurement in the table are tagged on the window on the left half of the GUI. Fibers with measurement values for the CPI or amplitude, or both lying out of the tolerance range have numbers which are colored differently. A red number

means that the fiber has its amplitude out of range, a blue number means that the fiber has its CPI out of range, and a magenta number means that the fiber has both its amplitude and CPI out of range.

G. Budget

The project was allotted a budget of $1500. As we already had access to MATLAB, the budget was only spent on building the fixture. A portion was spent on the camera itself, which required the need for a secondary lens. Purchasing optomechanical parts from Thorlabs made up the bulk of the spending, bringing the total up to $920.38, keeping us well below the budget.

Table 1 below shows exactly how the funds were distributed between the optics and the structure.

Table 1. Final Budget

Product	Quantity	Total Price
Trip to Monahan	1	$60.11
Camera	1	$309.98
38.1mm lens, 1” Dia	1	$37.00
LED Ring	1	$24.70
Aluminum Breadboard	1	$86.75
6” pedestal, 1” Dia	1	$43.00
6” extension, 1” Dia	1	$30.25
6” post, ½” Dia	5	$33.85
Clamping fork	1	$8.95
90 Angle Clamp	4	$39.04
Adjustable lens tube, 1” Dia	1	$20.35
Mounting ring	1	$18.30
24” structural rails	4	$116.00
Black hardboard, 3 sheets	1	$64.00
Screws	2	$15.10
Spraypaint	3	$13.00
Total		$920.38
Remaining budget		$570.62

IV. Conclusions and Recommendations

We successfully designed and constructed a fixture able to capture high quality images of multiple fibers. In addition, we developed an algorithm that accurately segments the images to instantaneously measure the diameter, amplitude, and CPI of each fiber. We also made a simple GUI to run the program. The program is an executable file and therefore has the flexibility to run on several Windows operating systems (7, 8, and 10), even those without MATLAB software. In addition, there are instructions provided for reference on how to install all the necessary software.

The success of this project can firstly be attributed to our ability to work as a team and split up the two major components. Though we had one team for the fixture and one team for the algorithm/GUI, we communicated constantly and gave input to each other to ensure that both teams were on the right track. Our clear problem statement and initial goals also allowed ample time to design and test the fixture. Using off-the-shelf parts as opposed to manufacturing our own saved us time and increased the overall quality of the project, without going over budget. Finally, the algorithm development went smoothly because we had access to MATLAB software, and tray iteration was possible due 3D printer access in our Mechanical Engineering Laboratory..

We would like to acknowledge Dr. Toussaint, who provided us with access to his lab and his graduate student, Woowon Lee. Their help was invaluable because the lab has an abundance of optomechanical parts which we used to test possible fixture setups. Additionally, Dr. Toussaint consistently provided support and advice which motivated us to create the highest quality project possible.

Based on our results, we recommend that a camera with autofocus and more standard focal length is used, removing the need for a secondary lens. We purchased the AmScope camera and were initially displeased with its functionality, however in the interest of time we continued to design our fixture around it and were fairly successful. Using a better camera could produce higher quality images while decreasing the fixture height and cost. Minor adjustments also need to be made to the GUI such that it fits the screen resolution of Monahan's computers. If possible, image acquisition should be implemented natively into the GUI to streamline the entire process.

Lastly, should the code ever need to be modified for other types of fibers, we are willing to provide future support over the summer for Monahan Filaments.

V. Appendix

Table 2. Error Calculations in Sample Fibers

Thickness (in)			CPI			Amplitude (in)		
Measured	Computed	%error	Measured	Computed	%error	Measured	Computed	%error
0.014	0.015007	7.1929	5	4.7495	-5.01	0.025	0.023	-8
0.0135	0.017735	31.3704	5	4.7469	-5.062	0.025	0.0195	-22
0.014	0.016371	16.9357	5	4.7469	-5.062	0.025	0.0214	-14.4
0.0135	0.013643	1.05926	5	4.7443	-5.114	0.025	0.0235	-6
0.014	0.015007	7.1929	5	4.7469	-5.062	0.025	0.0185	-26
0.0145	0.016371	12.9034	5	4.752	-4.96	0.025	0.0218	-12.8
0.0135	0.013643	1.0592	5	4.7572	-4.856	0.025	0.0238	-4.8
0.013	0.017735	36.4231	5	4.7752	-4.496	0.025	0.024	-4
0.0145	0.016371	12.9034	5	4.8677	-2.646	0.025	0.0235	-6

Main MATLAB code for image processing

```
I2 = rgb2gray(I2);% reads RGB image to grayscale
I2 = medfilt2(I2);
I2 = imbinarize(I2);
I2 = imcomplement(I2); % reverses black and white
if (strcmp('clear',clear_or_opaque))
      I2 = imcomplement(I2);
end
I2 = bwareaopen(I2,500);
angle = regionprops(I2,'Orientation'); %finds orientation of fiber
actual_angle = angle.Orientation;
if actual_angle == 0
      I2 = imrotate(I2, -90);
else
      I2 = imrotate(I2, -actual_angle);    %rotates image so that fiber is
horizontal
end
I2 = imfill(I2,'holes'); % fills up holes, not really necessary
r1 = regionprops(I2, 'BoundingBox','Area');
r1 = r1([r1.Area] > 200); %// Detect cells larger than some value.
I2 = imcrop(I2,[r1.BoundingBox]);
[height, length] = size(I2);
```

```
pixel_to_length = 1 / length;

%  Add cropping line for fibers here
if (strcmp('clear',clear_or_opaque))
      I_fiber = imcrop(I_color,[940 1200 3300 1350]);
      I = rgb2gray(I_fiber);   % reads RGB image to grayscale
      I = medfilt2(I);
      %I = imbinarize(I);
      I                                                                    =
imbinarize(I,'adaptive','ForegroundPolarity','bright','Sensitivity',0.4);
      I = bwareaopen(I,500);
      % axes(handles.axes2);
      % imshow(I);
else
      I_fiber = imcrop(I_color,[820 780 3300 1350]);
      I = rgb2gray(I_fiber);   % reads RGB image to grayscale
      I = medfilt2(I);
      %I = imbinarize(I);
      I                                                                    =
imbinarize(I,'adaptive','ForegroundPolarity','dark','Sensitivity',0.45);
      I = imcomplement(I); % reverses black and white
      I = bwareaopen(I,500);
      % axes(handles.axes2);
      % imshow(I);
end

r = regionprops(I, 'BoundingBox','Area');
s = regionprops(I, 'Centroid');

%r(1) = []; %// Clear 1st entry as it's the outside rectangle.

MaxArea = 1000; %// Select smallest area you want to keep.
r = r([r.Area] > MaxArea); %// Detect cells larger than some value.

L =  numel(fieldnames(r));
thickness_array = zeros(L,1);
amplitude_array = zeros(L,1);
period_array = zeros(L,1);

thickness_array_inch = zeros(L,1);
amplitude_array_inch = zeros(L,1);
CPI_array = zeros(L,1);

%mkdir individual_fibers_red
%mkdir extracted_wave_red
%mkdir fft_of_waves_red

I_color = imread(fullpathname);
```

```
imshow(I_color);
hold on;
for k = 1:L


      extractedcell=imcrop(I,[r(k).BoundingBox]);
      angle = regionprops(extractedcell,'Orientation'); %finds orientation of
fiber
      actual_angle = angle.Orientation;
      if actual_angle == 0
      extractedcell = imrotate(extractedcell, -90);
      else
      extractedcell = imrotate(extractedcell, -actual_angle);  %rotates image
so that fiber is horizontal
      end
      extractedcell = imfill(extractedcell,'holes'); % fills up holes, not
really necessary

      %     imshow(extractedcell);

%     imwrite(extractedcell,[pwd                    '/individual_fibers_red/fiber
',num2str(k),'.png']);

      height = size(extractedcell,1);
      width = size(extractedcell,2);

      top_boundary = zeros(1,width);
      bottom_boundary = zeros(1,width);

      for i = 1:width
      for j = 1:height
            if extractedcell(j,i) == 1;
            top_boundary(i) = j;
            end
      end

      for l = height:-1:1
            if extractedcell(l,i) == 1;
            bottom_boundary(i) = l;
            end
      end
      end

      averaged_wave = (top_boundary + bottom_boundary) / 2;
      oneD = averaged_wave(averaged_wave~=0);
      %oneD_smooth = oneD;
      oneD_smooth = smooth(oneD);
```

```
      oneD_smooth = oneD_smooth - mean(oneD_smooth); %shifts signal down to a
mean value of zero

      thickness = top_boundary - bottom_boundary;
      thickness = thickness(thickness~=0);
      thickness = round(mean(thickness)); %thickness in pixels, rounded to
nearest integer
      thickness_array(k) = thickness;

      Y = fft(oneD_smooth);
      % treat horizontal axis as "time" for the moment
      T = 1;
      Fs = 1;
      L = size(oneD_smooth,1);
      P2 = abs(Y/L);
      P1 = P2(1:L/2+1);
      P1(2:end-1) = 2*P1(2:end-1);

      f = Fs*(0:(L/2))/L; %in terms of cycles/pixel

      [amplitude_array(k),index] = max(P1(3:end));
      period_array(k) = 1 / f(2+index);

      thickness_array_inch(k) = thickness_array(k) * pixel_to_length;
      amplitude_array_inch(k) = (amplitude_array(k) * pixel_to_length) +
thickness_array_inch(k);
      CPI_array(k) = 1 / ((period_array(k) * pixel_to_length)/2);

      textcolor = 'k';
      error_flag = '';

      if (0.15*nominal_amplitude <= 0.005)
      if     ((amplitude_array_inch(k)     >     1.15*nominal_amplitude)     ||
(amplitude_array_inch(k) < 0.85*nominal_amplitude))
            error_flag = strcat(error_flag,'amp');
      end
      elseif   ((amplitude_array_inch(k)   >   0.005   +   nominal_amplitude)   ||
(amplitude_array_inch(k) < nominal_amplitude - 0.005))
      error_flag = strcat(error_flag,'amp');
      end


      if ((CPI_array(k) > nominal_CPI + 1) || (CPI_array(k) < nominal_CPI - 1))
            error_flag = strcat(error_flag,'cpi');
      end


      if (strcmp(error_flag,'amp'));
```

```
        textcolor = 'r';
        elseif (strcmp(error_flag,'cpi'));
        textcolor = 'b';
        elseif (strcmp(error_flag,'ampcpi'));
        textcolor = 'm';
        end

        if (strcmp('clear',clear_or_opaque))
        c = s(k).Centroid;
        text(c(1)+890, c(2)+1200, sprintf('%d', k), ...
              'HorizontalAlignment', 'center', ...
              'VerticalAlignment', 'middle', ...
              'FontSize', 12, ...
              'Color', textcolor);
        else
        c = s(k).Centroid;
        text(c(1)+770, c(2)+780, sprintf('%d', k), ...
        'HorizontalAlignment', 'center', ...
        'VerticalAlignment', 'middle', ...
        'FontSize', 12, ...
        'Color', textcolor);
        end
end
hold off;

thickness_mean = mean(thickness_array_inch);
thickness_std =  std(thickness_array_inch);

amplitude_mean = mean(amplitude_array_inch);
amplitude_std =  std(amplitude_array_inch);

CPI_mean = mean(CPI_array);
CPI_std =  std(CPI_array);

filename_thick = ['Thickness, ',datestr(now, 'mm-dd-yyyy, HH-MM'),'.csv'];
filename_amp = ['Amplitude, ',datestr(now, 'mm-dd-yyyy, HH-MM'),'.csv'];
filename_cpi = ['CPI, ',datestr(now,'mm-dd-yyyy, HH-MM'),'.csv'];


xlswrite(filename_thick, thickness_array_inch);
xlswrite(filename_amp, amplitude_array_inch);
xlswrite(filename_cpi, CPI_array);
```