An Image Analysis Pipeline to Extract Diameter Measurements from 3D Muscle Fiber Images

by

Dylan John Mendonca

A thesis submitted in conformity with the requirements
for the degree of Master of Engineering

Department of Chemical Engineering and Applied Chemistry
University of Toronto

An Image Analysis Pipeline to Extract Diameter Measurements from 3D Muscle Fiber Images

Dylan John Mendonca

Master of Engineering

Department of Chemical Engineering and Applied Chemistry
University of Toronto

2020

Abstract

The discovery of novel therapeutics that stimulate muscle regeneration has the potential to improve the lives of patients suffering from muscular degenerative conditions. To aid in this effort, the Gilbert and McGuigan labs collaborated to develop a 3D *in vitro* **M**uscle **End**ogenous **R**epair (MEndR) model, which can recapitulate fiber morphology and help stratify drugs based on their ability to enhance muscle repair. Recent improvements in fabrication and imaging in MEndR have increased its throughput, enabling researchers to collect many more 3D images for analysis. Despite these extensions, MEndR is still limited by the speed of manual annotation. To alleviate the bottlenecks in analysis, this report aims to design an automated pipeline that extracts the distribution of muscle fiber diameters from 3D MEndR images. The study first presents an orthogonal projection and outlier detection algorithm (OPlanaR) that measures the diameter of thresholded fibers; and assesses its feasibility using artificially generated fiber networks. Next, the OPlanaR algorithm is implemented within an end-to-end image analysis pipeline and its performance is evaluated against manually annotated MEndR images. The OPlanaR algorithm successfully analyzes artificial images, with an error of ~3.5% against the true diameter. The image analysis pipeline performed well on MEndR images, with a ~6.4% discrepancy against the mean diameter measured through manual annotation. However, the findings suggest that the pipeline might not perform as well on images with dense fiber networks. To address this limitation, future work should aim to address the systematic errors associated with dense networks, validate different stages of the pipeline, and improve the outlier detection model using machine learning-based approaches. Nevertheless, this work highlights the value of automated pipelines in increasing the throughput of MEndR analysis; taking us one step closer to a platform that can quickly identify promising therapeutic avenues that enhance muscle repair.

Acknowledgments

Thank you to Professor Alison McGuigan for allowing me to work on this project, for teaching me how to formulate a research question, and how to communicate effectively.

Thank you to Jose and Amir for mentoring me, supporting me, helping me with the coding aspects, and introducing me to the wonderful world of graph theory (the most beautiful data structure ever).

Thank you to Nancy for teaching me how to draw figures, structure thoughts formally, and to navigate the research space.

Thank you to Vera and Vincenzo for supporting me mentally during this entire process.

Thank you to the rest of the members of the McGuigan lab: Simon, Nila, Ileana, Michael, and Natalie for welcoming me into the lab and inspiring me to work harder.

Thank you to my family and friends for being a beacon of support, guidance, and love. Although it has been a tough year, you all have motivated me to stay on track, follow my dreams, and commit right up to the end – these are valuable lessons that I will forever cherish.

Table of Contents

1 Introduction

1.1 The Computational Paradigm for 3D Biological Models

There has been a marked increase in the popularity of 3D biological models over the last few years, due to their ability to more accurately simulate physiological processes *in vitro* compared to their 2D counterparts [1]. These developments, combined with rapid advances in sample preparation and 3D optical imaging methods, including Light-Sheet Fluorescence, confocal, and multi-photon microscopy, have generated vast amounts of image data for analysis [2]. The quantification of this information-rich data could unlock many insights into the function of biological systems and enable more rapid advances in drug discovery; if it were not for the difficulty and time required for manual analysis. Computerized systems can aid in this regard, by offering researchers the ability to take quick, repeatable measurements of images, thus relieving them of the burden of manual annotation [3]. Furthermore, computational approaches could allow scientists to discover new properties of cellular systems, that would have been otherwise difficult to investigate using manual methods.

Despite the growth in 3D image data and the availability of open-source algorithms for 3D image analysis and visualization, there have been very few attempts to analyze 3D cellular images in the biological space [2]. This can be attributed, in part, to the variety of image datasets, the diversity of research questions, and the inaccessibility of complex 3D algorithms to biologists [2, 4]. That being said, there is a vast body of research in applying algorithms to 2D images, which can be exploited to the 3D setting [2, 5].

1.2 Motivation & Aims

This report will focus on a promising 3D **M**uscle **End**ogenous **R**epair (MEndR) *in vitro* model, that is capable of stratifying compounds based on their capacity to enhance muscular repair. Meysami Fard et al. recently extended this model, by increasing its throughput to a 96-well capacity, and developing a pipeline to more rapidly generate 3D images of the muscle tissue [6].

The extensions created by Meysami Fard et al. have the potential to increase MEndR's throughput and generate many more image samples for analysis. However, MEndR is still limited by the speed at which researchers can manually analyze the images. Automated pipelines that can extract information about the spaciotemporal dynamics of fibers produced in MEndR could help

alleviate the bottleneck in analysis and help researchers find promising muscle regenerative therapeutic candidates more quickly. The analysis of morphological features of fibers including diameter, length, and count is of particular interest, because these features of a muscle fiber's cytoplasm are important biomarkers for muscle health [7, 8]. For example, an algorithm that could count the number of fibers in an image could have enormous implications on the speed at which a research group can investigate the impact of different drug candidates on muscle regeneration, by allowing scientists to more quickly count the number of muscle fibers before and after drug administration.

Given the necessity for computational image processing methods for MEndR, this report focuses on the development of an automated image analysis pipeline that can measure the distribution of muscle fiber diameters from a 3D image generated by the platform. The primary challenge associated with realizing this objective is that fibers in the image appear to be fused and are often unaligned, making the quantification of geometry difficult (Figure 1) [9]. The appearance of fused fibers can be attributed to resolution limits and the point spread function of microscopes, while the unalignment is a consequence of how fibers grow in MEndR. To address this challenge, the report aims to implement an orthogonal projection strategy and outlier classification system that can measure the diameter distribution of segmented fibers. It then evaluates the performance of the algorithm using artificially generated fiber networks. The report then integrates this algorithm into an end-to-end image analysis pipeline and applies it to a series of images produced by MEndR. Finally, the study validates the performance of the pipeline by comparing the model's diameter predictions to those measured by manual annotation.

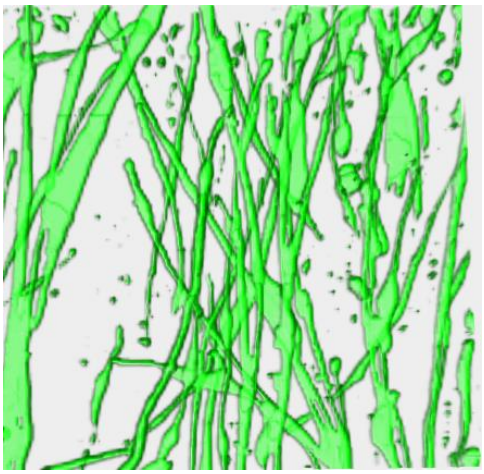


Figure 1: Top-view of a muscle fiber image produced in MEndR, demonstrating that muscle fibers appear to be fused and are very unaligned.

2 Literature Review

The subsequent sections provide a literature review of MEndR and of image analysis pipelines applied to a variety of tissue models (including muscle fiber models). Section 2.1.1 gives readers a brief overview of the biological model considered and its value as a screening platform. Section 2.1.2 will help orient readers to the typical framework applied to image analysis tasks and gain a grasp for different approaches that have been attempted on 2/3D tissue models. Finally, Section 2.1.3 gives readers a review of image processing approaches applied to muscle fibers, given the biological focus of this report.

2.1.1 Muscle Endogenous Repair Model (MEndR)

Muscle **E**ndogenous **R**epair model (MEndR) is a 3D *in vitro* biological model created by the Gilbert and McGuigan labs that can recapitulate *in vivo* muscle endogenous repair processes. This model was created to address the time and financial constraints associated with performing *in vivo* tests on muscle fibers, and to supplant the lack of *in vitro* systems that can simulate muscle repair.

The MEndR model consists of primary muscle stem cells (myoblasts) that are engrafted on a thin sheet of muscle fibers and cultured on cellulose scaffolds, which are then injured using cardiotoxin [6]. The tissue samples are left to regenerate and can be imaged after the culture period. This model was shown to accurately predict compounds that could enhance muscle endogenous repair, by comparing its results to an *in vivo* transplantation assay (the gold-standard for testing regeneration therapies) [6]. Apart from its predictive ability, MEndR was shown to accurately portray the spaciotemporal dynamics of muscle fibers in images and had strong compatibility with high-content screening methods, making it an attractive biological model to investigate muscle repair [6]. However, the MEndR model was severely limited by its throughput, hindering its use as a screening platform.

Meysami Fard et al. recently addressed the bottlenecks related to fabrication and imaging in MEndR, by increasing its throughput to 96-well capacity and developing methods to generate 3D images more easily [6]. These extensions have meant that MEndR can now generate a vast number of images that capture the spaciotemporal dynamics and properties of muscle fibers. However, the platform is limited by the speed of analysis, given the challenges of manually analyzing images in 3D. Automated image analysis pipelines can help reduce this bottleneck, by

allowing researchers to take quick, repeatable measurements of fiber properties within these images.

2.1.2 Image Analysis Pipelines

The development of image analysis pipelines for biological models has led to the creation of a generalized image analysis workflow (Figure 2), which typically consists of the following steps: (i) pre-processing to enhance contrast in the image (ii) segmentation to partition the cells/tissue of interest from the background, (iii) processing to orient the image for analysis, and (iv) analysis to extract relevant information [10, 11]. The subsequent sections briefly review common challenges encountered within each of these steps, and a few approaches that have been adopted to address these problems. It should be noted that in practice, a single algorithm, sequence, or combination of algorithms may be applied for each step.

2.1.2.1 Pre-Processing

Pre-processing is an important preliminary step in an image analysis workflow as it helps alleviate uneven illumination, low resolution, low contrast, and high noise, which are inherent biases that arise from image acquisition [3, 5, 10, 12]. This step is very data-dependent, and requires parameter tuning to avoid under- or over-segmentation in downstream image analysis steps [10]. Most pre-processing approaches fall into two buckets, which include: (i) applying filters to images (Laplace of Gaussian [5], Gaussian [9], Edge Preserving [13], Rolling Ball [14]), or (ii) manipulating the pixel intensity histogram of an image (Image Contrast Sketching [3], Histogram Matching [10], Normalization [15]).

2.1.2.2 Segmentation

Segmentation (or thresholding/binarization) helps partition the background from the cells/objects of interest. Algorithms in this step typically generate a binary mask of the image, where the background and foreground are represented by 0 and 1, respectively.

Segmentation approaches typically fall into classical or modern categories. Classical methods include global thresholding (Hard Constraint [9], Otsu's Method [16]) and adaptive thresholding* (Local Hard Constraint [7], Local Otsu [17]). Global thresholding approaches are very common in literature because they are easy to implement and are available on several open-source platforms [15]. However, they can be quite limited in images with non-homogenous

backgrounds, as they are not very effective in capturing regional context [3]. Adaptive methods can aid in this regard but are not as robust as modern approaches [15].

Modern approaches include graph-cuts [3, 18], traditional machine learning techniques ([Random Forest Classifiers [19], Support Vector Machines [20]), and deep learning methods (U-Nets [4, 10, 15, 21], Convolutional Neural Networks [22], Convolutional Neural Network + Recurrent Neural Network [23]). Graph-cuts techniques for image segmentation are based on graph theory. They are very robust and suitable for difficult segmentation tasks with complicated objects because they can incorporate both regional and global information simultaneously [3, 24, 25]. Traditional machine learning methods for segmentation have been integrated successfully into openly accessible tools such as WEKA Segmentation in ImageJ [19] and ilastik [20]. They are very useful for segmentation in the 2D setting but fail to capture global context in the 3D setting because segmentations on these platforms are done on a slice-by-slice basis. Moreover, traditional machine learning models are less effective at segmentation compared to their deep learning counterparts, especially when it comes to objects with occluded boundaries [21, 23, 26]. While deep learning models are state-of-the-art and have proved to outperform classical methods by a large margin, they are prohibitive for users because they are data-hungry, difficult to implement, and require manual painting in 3D [15, 22].

*Adaptive thresholding methods are a special case of combining a local filter (pre-processing step) with a global thresholding technique (segmentation step)

2.1.2.3 Processing

The goal of the processing (or refinement) step is to make the image amenable for downstream analysis. Hence, this step is very problem/data specific. The processing stage of the pipeline addresses a wide variety of challenges, including, but not limited to, the separation of fused objects [3, 5, 17, 18], the identification of distinct objects [4], the removal of artifacts [7], the extraction of an alternative representation of the objects [9], and the refinement of segmented objects [10]. There are many methods that apply within this category, including morphological operations (small object removal, hole filling, etc.) [7], shape-fitting models [5], skeletonization [9], watershed approaches [7, 10, 17, 27], level-set methods [28], clustering [29], conditional random fields/graph-based methods [10, 18], machine learning approaches [17, 27], and deep learning architectures [30].

2.1.2.4 Analysis

This step typically involves extracting features from the image into a dataset for statistical analysis to answer a research question. Common features of cells/tissues that are extracted for analysis include diameter, length, count, and shape.

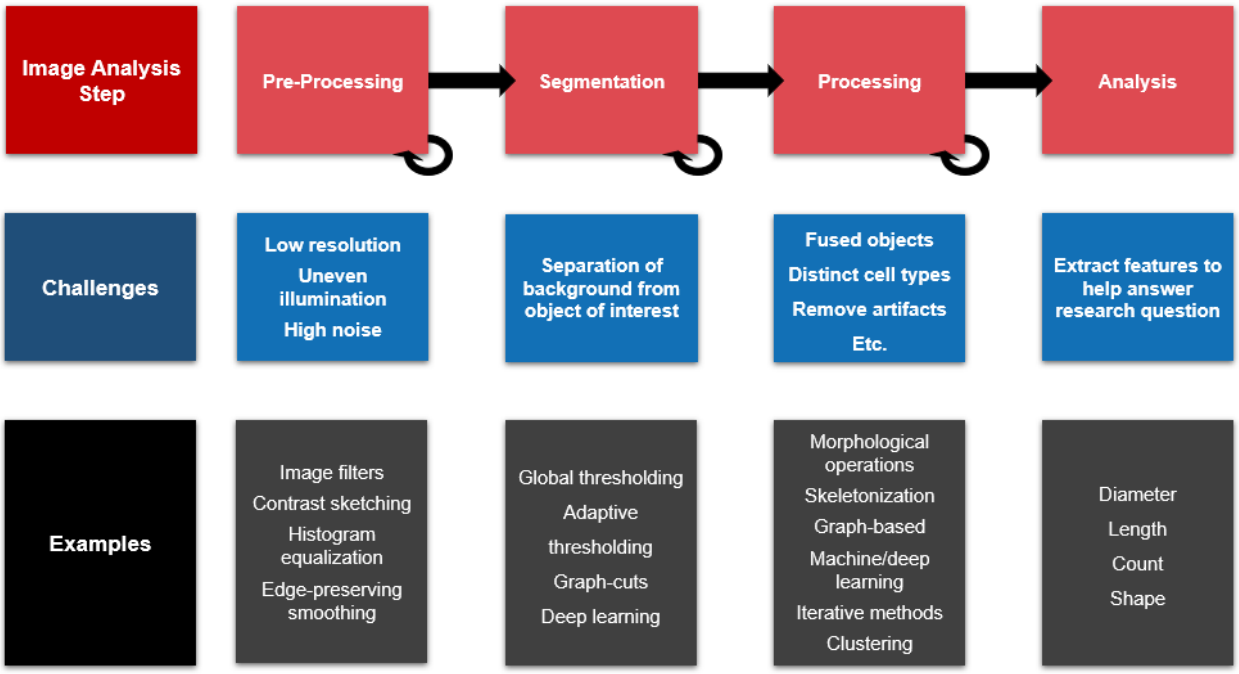


Figure 2: The framework for image analysis

2.1.3 Image Analysis on Muscle Fibers

There has been no compelling approach to analyze the morphology of muscle fibers in 3D. However, there have been many 2D computational approaches to identify distinct skeletal muscle fibers for analysis. For example, work by Janssens et al. used a supervised classifier to identify clumps of muscle cells and then applied an iterative concavity splitting technique to split clumps [31]. A paper by Su et al. proposed an ellipse fitting, refinement, and mean-shift clustering method based on geodesic distance to separate and label unique muscle fibers [29]. Miazaki et al. implemented a stepwise method of morphological operations to incrementally detect and segment muscle fibers by using convexity analysis [8]. Mula et al.'s pipeline enhanced the boundaries of fibers using ridge detection, localized seeds using concave area identification, and then applied a gradient vector flow model to delineate fiber boundaries [32]. Liu et al. implemented a boosting-based machine learning method to detect muscle fiber seed points, and then used a gradient balloon snake model to generate the boundaries of the muscle fibers [33]. While these groups created

highly accurate and precise tools to separate unique muscle fiber boundaries, their datasets were based on 2D cross-sections of muscle fibers and were often tested on *in vivo* biological models.

There have been very few approaches to detect, segment, and analyze muscle fibers along a longitudinal direction. Comin et al. presented an image processing pipeline consisting of thresholding, small hole filling, skeletonization, and pruning to detect distinct muscle fibers for diameter, count, and shape analysis [7]. Guo et al. developed a pipeline that consisted of image contrast sketching, constrained graph cuts segmentation, hole filling, and an iterative splitting technique to segment unique muscle fibers for analysis [3]. Both approaches performed very well compared to manual annotation but once again, were restricted to the 2D setting.

3 Approach

All the algorithms discussed in this section were created on MATLAB version R2019b. Information on locating the source code for the final MATLAB implementation of the project and the preliminary Python implementation can be found in Appendix A.

3.1 Model Development

The primary challenge in this study was that fiber objects appear fused and are often unaligned. Hence, significant effort was devoted to the development of a 3D orthogonal projection strategy and outlier detection system to measure the diameter of fibers. This “processing” algorithm (referred to as the OPlanaR algorithm for the remainder of the report) was evaluated on artificially generated networks of fibers. Artificial images were used as a proof-of-concept and to test the algorithm before applying it to real images. The following sub-sections outline the methods used to generate artificial networks, the steps in the OPlanaR algorithm, and the experiments conducted to evaluate the algorithm.

3.1.1 Artificial Network Generation

A 3D image of randomly positioned cylinders was used to simulate images of muscle fibers produced in MEndR. Cylinders were chosen as the default shape they most accurately represented the actual shape of a skeletal muscle fiber. The artificial network was a binary image (fibers and background represented by 1 and 0, respectively), based on the assumption that the network had already been segmented from the background.

Four parameters were specified to create an artificial network: the image length/width (*S*), image depth (*Z*), number of cylinders (*N*), and cylinder radius (*R*). First, an empty 3D image of size [*S*, *S*, *Z*] was created. The spine of a cylinder was specified by selecting two random points in the image and generating a set of points that connected the two with a straight line. A cylinder was specified by computing the Euclidean distance of all voxels in the image to each voxel of the spine and setting the intensity of those voxels within a radius of *R* to a value of 1. The process was repeated *N* times to generate the final network. A sample network has been included in Figure 3.

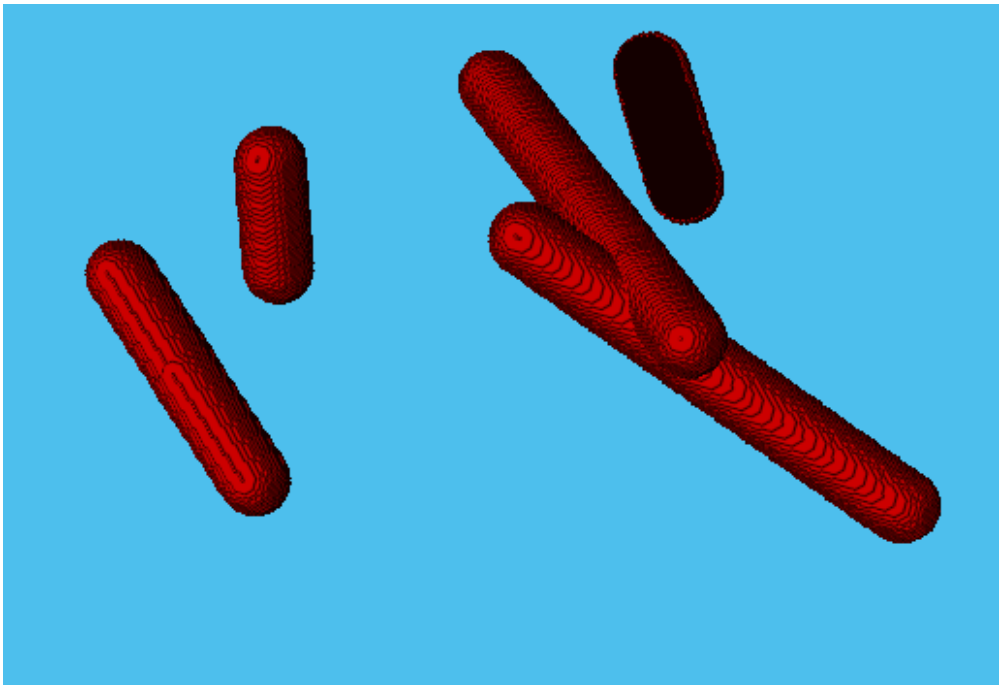


Figure 3: Sample of an artificial network. The artificial network was created using *S*=300, *Z*=100, *N*=5, *R*=15

3.1.2 OPlanaR Algorithm

An orthogonal projection strategy and outlier detection model was then applied to extract the diameter distribution of artificial fibers in the image. The orthogonal projection approach was chosen over taking purely planar cross-sections of the image because it would yield more accurate cross-sectional representations of fibers (Figure 4). Furthermore, since fibers are not aligned and sometimes overlap, it would have been very challenging to determine the single optimal plane to cut the image.

To simplify the problem, chunks of disjoint fibers were separated from each other based on connectivity to yield a set, *O*, of distinct connected components o_i, where $O = \{o_1, o_2 \dots o_n\}$. The steps described in the following sub-sections were applied to each distinct fiber clump, o_i.

Each connected component, o_i, consists of a set of points $b_{i,j}$ associated with that object, where $o_i = \{b_{i,1}, b_{i,2}, \ldots, b_{i,j}\}$. A visual representation of the algorithm formulation (which has also described in the following sub-sections) has been included in Figure 6 as a reference.

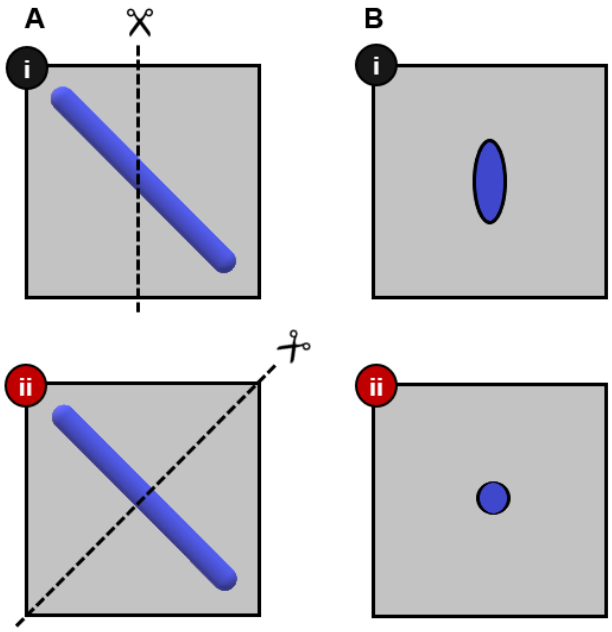


Figure 4: Advantages of orthogonal projections over planar cross-sections. (A) 3D schematic of (i) Taking a cross section that is not orthogonal to the alignment of fiber (ii) taking a cross-section of a fiber that is orthogonal to fiber alignment. (B) Visualization of cross section from (A i) and (A ii) where (B i) would result in an overestimation of fiber diameter while (B ii) would yield a more accurate representation.

3.1.2.1 Skeletonization

The 3D object was skeletonized using MATLAB's *bwskel* function to extract a one-pixel representation of its medial axis [34]. The skeleton was then converted into a graph network structure using the *Skel2Graph3D* function from MATLAB's file exchange [35].

The graph structure of the skeleton network $N = \{V, L\}$ consists of a set of m vertices $V = \{v_1, v_2, \ldots, v_m\}$ and n links $L = \{l_1, l_2, \ldots, l_n\}$. Each vertex v_i is represented by a set of 3D coordinates (x_i, y_i, z_i). Each link l_i consists of a start vertex, end vertex, and a set of vertices between the two. For example, the link, l_1, could be the line connecting the start and end vertex, v_1 and v_4, respectively, with the vertices, v_2 and v_3 in between. Figure 5 provides a visual aid to understand the graphical structure of a skeleton network.

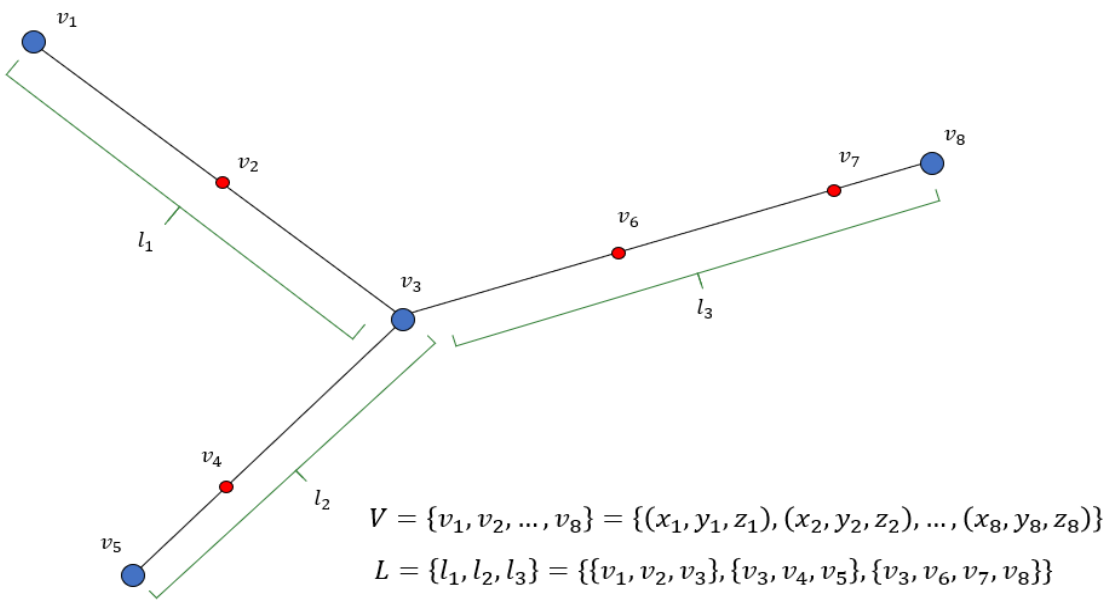


Figure 5: Visual depiction of a 3D skeleton graph network with 8 vertices and 3 links. Start and end vertices of a link have been represented by blue circles. Intermediate vertices have been represented by red circles.

3.1.2.2 Gaussian Smoothing and Gradient Calculation

For each link in the skeleton, the 3D coordinates of the vertices in the link was extracted and smoothed using a Gaussian smoothing function, *gsmooth*, obtained from MATLAB's file exchange [36]. In this report, a 1D Gaussian filter with a fixed standard deviation (σ) was applied to the set of *x*, *y*, and *z* coordinates separately to obtain the smoothed set of *x*, *y*, and *z* coordinates. The equations to compute the Gaussian filter have been included in Appendix B.

The tangent vectors at each smoothed vertex in a link were computed by applying a centered difference filter of size [3,1] to each smoothed vertex in a link using the *gradients_x* function obtained from MATLAB's file exchange [36]. The centered difference filter (Appendix B) is equivalent to taking the difference between two vertices to the right and left of the vertex of interest. This yields a vector that is tangent to the skeleton at that vertex. Each gradient vector was normalized and converted into a unit vector.

3.1.2.3 Orthogonal Projections

Orthogonal projections along the network skeleton were used to obtain representative fiber cross-sections for morphological analysis. For each smoothed vertex u_k in each link, all the points in

object o_i that fell within a sphere of radius *rMax* were extracted to generate a set of points B_k (Equation 1).

Equation 1: Generating a set of points B that contain all voxels with a sphere of radius rMax from the smoothed vertex

$$B_k = \{b_{i,j} \in o_i \mid \|u_k - b_{i,j}\|_2 < rMax\}$$

$$where \ \|*\|_2 \ is \ the \ Euclidean \ distance \ between \ the \ two \ points$$

The distance of each voxel in B from the theoretical 2D orthogonal plane at the smoothed vertex u_k was computed by taking the dot product of: (i) the vector between the voxel $b_{i,j}$ and the smoothed coordinate u_k, and (ii) the unit gradient vector ∂u_k (Equation 2). The points that fall within a specified distance, *d2plane*, were extracted to obtain a final set of points (P_k) that lie on the theoretical 2D projection at u_k (Equation 3).

Equation 2: The distance to the theoretical orthogonal plane at u_k is computed using a dot product

$$d_{i,j} = (b_{i,j} - u_k) \cdot \partial u_k$$

Equation 3: Points that are within a distance d2plane of the orthogonal plane are kept as the projection

$$P_k = \{p \in B_k \mid d_{i,j} \leq d2plane\}$$

The final set of coordinates in P_k are 3-dimensional. To permit visualization and extract properties of the projection, a linear dimensionality reduction technique, called Principal Component Analysis (PCA), was applied to the set of 3D coordinates. The PCA was done using the *pca* function in MATLAB [34]. The top 2 principal components are extracted and used to convert the set of 3D coordinates into a 2D representation of the orthogonal projection. The set of 2D coordinates is then converted into an image data structure (array-based representation) to take advantage of the image analysis functions in MATLAB.

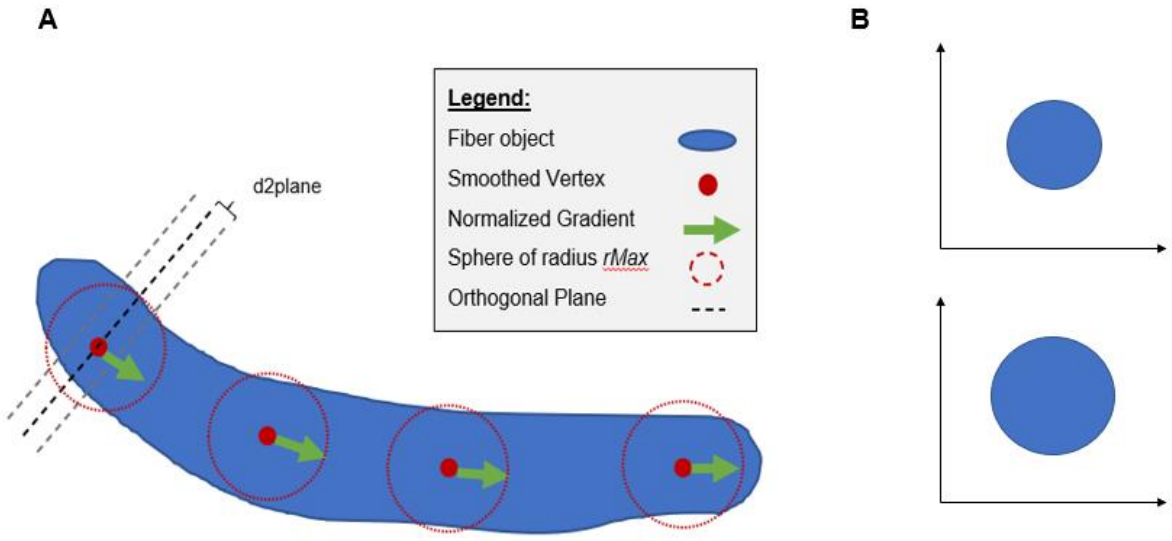


Figure 6: Visual Representation of OPlanaR algorithm. (A) A top view of a link for a fiber object has been drawn. The link consists of 4 vertices (red circles). The figure displays the smoothed vertices that were obtained after applying Gaussian smoothing. The gradient vectors for each smoothed vertex have been drawn with green arrows. The OPlanaR algorithm extracts all the blue voxels located within the sphere for a vertex and projects those points onto the theoretical orthogonal plane (black dashed line) using the dot product of the normalized tangent vector (green arrow) and the vector between the smoothed vertex and the blue voxel. Those projected points within a distance *d2plane* are retained as the orthogonal projection for a vertex. (B) Example projections that may be obtained from the fiber in (A). One link has many projections with different diameters and shapes.

3.1.2.4 Feature Extraction and Outlier Classification

The steps above yield a set of 2D image projections for all the vertices in every link of the skeleton for each fiber object in the image. The projections appear as blobs when displayed as an image. Occasionally, there may be more than 1 blob in a projection. The number of blobs in a projection depends on the location of the skeleton in relation to the object. Eight features (equivalent/major-axis/minor-axis diameter, perimeter, eccentricity, solidity, circularity, and convexity) were extracted for each disconnected blob in the projection image using the *regionprops* function [34].

While analyzing the 3D images along planar cross-sections during the preliminary stages of this study, it was observed that blobs had uneven shapes in areas of fiber overlap (Figure 7). A fundamental assumption in this work is that unevenly shaped blobs should not be measured for diameter because they do not adequately represent a disjoint fiber cross-section. Another assumption was that these unevenly shaped blobs would be produced using the orthogonal projection strategy. Hence, an outlier model was applied to each projection to delineate between regular and irregularly shaped blobs.

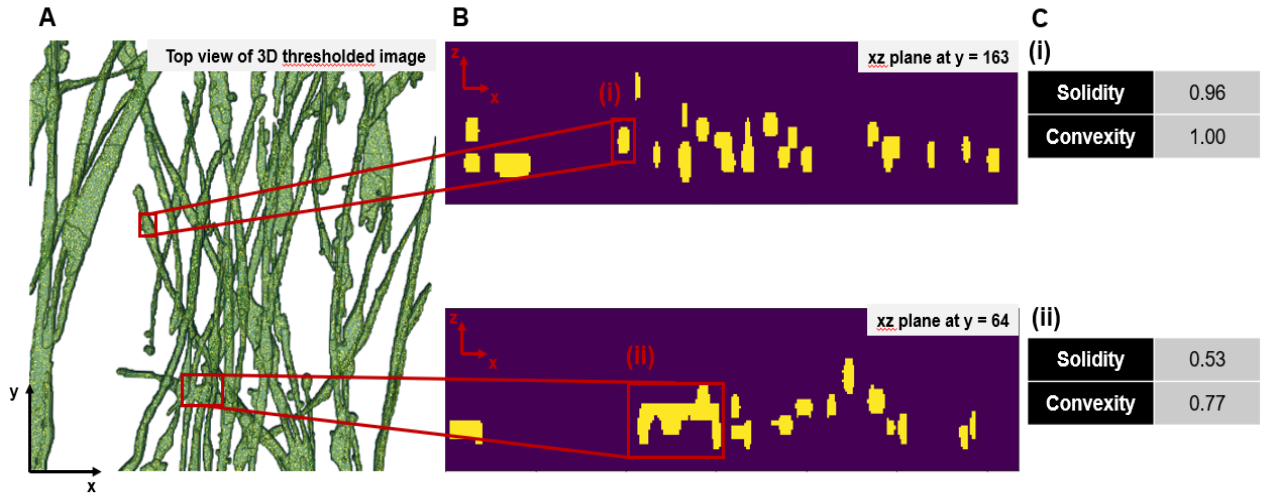


Figure 7: Motivation behind outlier model to select "measurable" blobs. (A) XY-planar view of a 3D thresholded MEndR image with small objects removed. (B) Cross-sections of: (i) a unique fiber along the xz-plane at y = 163, (ii) a cluster of overlapping, merged fibers along the xz-plane at the slice y = 64. (C) Features extracted from blobs (i) and (ii). A critical challenge faced was that measurements of merged blobs as shown in (B ii) would not yield reliable results compared to measuring blobs like (B i), which are more reasonable representations of the fiber. This observation motivated the development of a binary classifier that leverages the shape features described in (C) to select viable blobs for measurement.

The outlier model was built by analyzing the convexity and solidity of the blobs. Convexity helps measure the relative amount that the perimeter of an object differs from the perimeter of its convex hull (Equation 4), while solidity helps measure the amount that the area of an object differs from the area of its convex hull (Equation 5). The convex hull is the smallest polygon that encloses all points in a blob (Figure 8). Both shape features help indicate the "regularity" of an object; those objects with low convexity or solidity typically have irregular boundaries.

Equation 4: Function to compute convexity of a blob

$$Convexity = \frac{Perimeter}{Convex\ Perimeter}$$

Equation 5: Function to compute solidity of a blob

$$Solidity = \frac{Area}{Convex\ Area}$$

The outlier model sets a hard threshold for convexity (t_c) and solidity (t_s) and classifies any blobs with either convexity or solidity below their respective thresholds as an outlier. The distribution of equivalent diameter of the inlier blobs is finally extracted for experimental analysis. It should be noted that although equivalent diameter was extracted for this report, the OPlanaR model can measure any feature of a 2D shape, including minor axis diameter, major axis diameter,

perimeter, ellipticity, eccentricity, and so on. In addition, the outlier detection system in the OPlanaR algorithm can help indicate the macro structure of a fiber network. For example, networks with many outliers suggests that many clumps of fibers exist, indicating the presence of a well-connected muscle tissue.

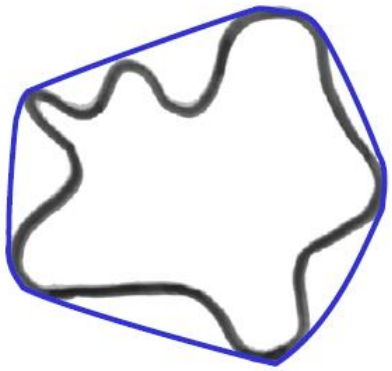


Figure 8: The convex hull is the smallest polygon that encloses a shape [37]

3.1.3 Experiments on Artificial Images

A sensitivity analysis was conducted to determine the impact of hyperparameters ($d2plane, t_c, t_s, \sigma, rMax$) on the model's final prediction for equivalent diameter. The sensitivity analysis was done to evaluate the performance of the algorithm and to determine an appropriate set of hyperparameters for future experiments. As part of the sensitivity analysis, five unique artificial networks of size [300,300,100] containing 5 cylinders of radius 15px were generated. A base set of algorithm hyperparameters was chosen. The algorithm was then applied to each artificial image by varying one hyperparameter from the base set and recording the outlier fraction and mean equivalent diameter output by the model. The mean equivalent diameter was compared to the true diameter of the artificial network (30px in this case). The set of hyperparameters tested have been included in Appendix C.

The artificially generated networks were also used to test the effect of increasing network density on model speed and accuracy. The density of an image was varied by changing the number of cylinders in the network. The pixel density of an image was then computed by dividing the count of "on" voxels by the total number of voxels in the image (Equation 6). The size of the image, radius of cylinders, and hyperparameters was the same as the base set used during the sensitivity analysis.

Equation 6: Pixel density is computed by dividing the number of "on" voxels by the total number of voxels

$$Image\ Density\ (\%) = \frac{\#\ of\ voxels\ equal\ to\ 1}{Total\ \#\ of\ voxels} * 100$$

3.2 Implementation of Model in End-to-End Pipeline

The OPlanaR algorithm was then implemented as the "processing" step in an end-to-end image analysis pipeline for 3D muscle fiber images produced by MEndR (Figure 10). Given the time constraints and the scope of the report, validation was not conducted on the pre-processing and thresholding stages of the pipeline. The goal of this section was to develop a preliminary pipeline that could be used as a starting point for future research. It should be noted that the OPlanaR algorithm operates on a binary mask and that its operation is independent of any other stage of the pipeline. Hence, any pre-processing or thresholding method can be used. However, the quality of the binarization will most likely impact the results returned by the model. The subsequent sub-sections briefly outline the pipeline and the experiments conducted with it.

3.2.1 Image Analysis Pipeline

Minimum-maximum (min-max) normalization was applied to each voxel (v) in the 3D image to squeeze the intensities of voxels to the [0,1] range. Min-max normalization is a linear transformation wherein the normalized intensity at a voxel ($I_{norm}(v)$) is computed by taking the difference between the intensity at that particular voxel ($I(v)$) and the minimum image intensity; and dividing that result by the difference between the maximum and minimum intensities in the image (Equation 7).

Equation 7: Min-max normalization applied to each voxel in the image

$$I_{norm}(v) = \frac{I(v) - \min(I)}{\max(I) - \min(I)}$$

Anisotropic diffusion (edge-preserving) smoothing was then applied to reduce image noise and smooth the image, while maintaining edges [13]. Edge-preserving smoothing is a non-linear image processing technique that applies diffusion-based partial differential equations (resembling physical transport phenomena) to an image over time. The smoothing equations penalizes diffusion between inhomogeneous regions (i.e. across edges or boundaries) and encourages diffusion within

homogenous regions, which has the effect of blurring an image while retaining edges. The smoothing function, *anisodiff3D*, was taken from the MATLAB file exchange [38].

A graph cuts-based segmentation with modified constraints was then applied to segment the fiber objects from the background. The graph-cuts algorithm borrows optimization algorithms from graph theory to separate the vertices of an image graph into disparate groups, by determining the minimum cut that will yield the maximum flow in the graph from the source (or foreground) node to the sink (or background) node (Figure 9). Graph-cuts is a very robust method for segmenting objects because it incorporates both local and global information [25].

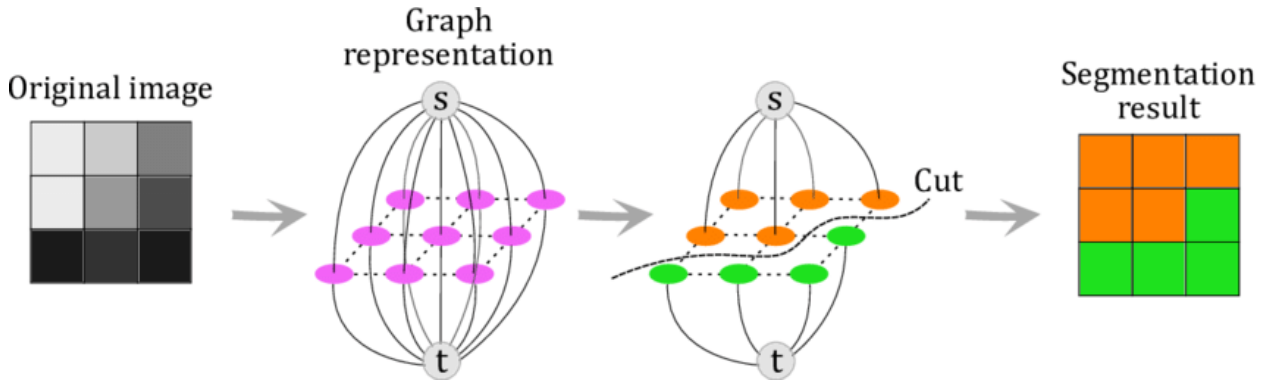


Figure 9: Graph cuts applied to 2D images. An image is converted into a graph where each pixel is connected to one another through edges, weighted according to the similarity of pixel intensities. The graph also contains a source (s) and sink (t) node, to which each pixel is connected using weighted edges. A minimum cut/maximum flow algorithm is then applied to determine the optimal cut i.e. the smallest total weight of edges that would disconnect the source from the sink. This graph is then converted back into an image, where those vertices connected to the s node are foreground and those connected to the t node are background [39].

The segmentation strategy in this report was adapted from Guo et al. from the 2D setting to 3D [3]. A 26-connected 3D graph of the image was generated using the *imageGraph3* function from the MATLAB file exchange [40]. Two terminal nodes (*s* and *t*) were initialized and connected to every vertex. Pixel-pixel and pixel-terminal node edge weights were specified using a Gaussian function and Radial Basis Function, respectively, with the same hyperparameters specified by Guo et al. A hard area-based constraint (the same as Guo et al.) was imposed while specifying the pixel-foreground/pixel-background edge weights, to account for nuclei and tiny artifacts in the image. A MATLAB implementation of the min-cut/max-flow partitioning algorithm, *maxflow*, was applied to the image graph to extract the vertices connected to the source from those connected to the sink [34]. Due to computational memory limitations, the 3D image was split into 16 equal parts, binarized individually using graph-cuts, and then stitched back together to yield the final segmented image.

Small holes in the image were then filled using MATLAB's morphological region filling function, *imfill* [34]. Objects with volumes less than 2000 voxels were removed using MATLAB's small object removal function, *bwareaopen* [34]. After resizing the image to account for voxel spacing, the OPlanaR algorithm described in Section 3.1.2 was applied to the image and the distribution of equivalent diameter measurements of the inlier projections was extracted for analysis.

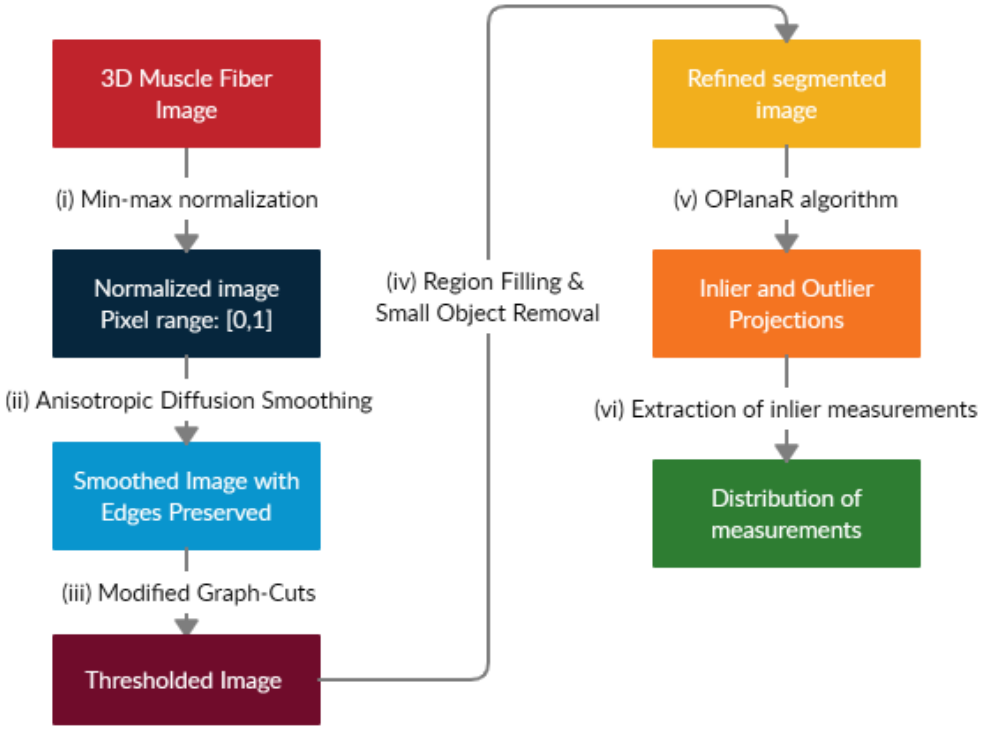


Figure 10: Overview of image analysis pipeline for 3D MEndR images. The pipeline consists of: (i) min-max normalization applied to 3D muscle fiber image, (ii) anisotropic diffusion (edge-preserving) smoothing to smooth the image (iii) modified graph-cuts segmentation to segment fibers from background (iv) region filling and small object removal to refine the image, (v) OPlanaR algorithm to extract orthogonal projections and classify them as inlier or outliers, (vi) extraction of equivalent diameters of inlier blobs for statistical analysis.

3.2.2 Experiments on MEndR Images

Four MEndR images with manually annotated diameters were taken from a student at the lab to help evaluate the performance of the pipeline. The images were based on 4 different trials of the same biological experiment. Each image contained 50 unique diameter measurements. The pipeline described in Section 3.2.1 was applied to each image and the equivalent diameter and pipeline processing time was extracted. A 2-sided Kolmogorov-Smirnoff statistical test was used to check whether the model's diameter distribution matched that of the manual annotator. In addition, the mean diameter predicted by the model and the manual annotator was compared.

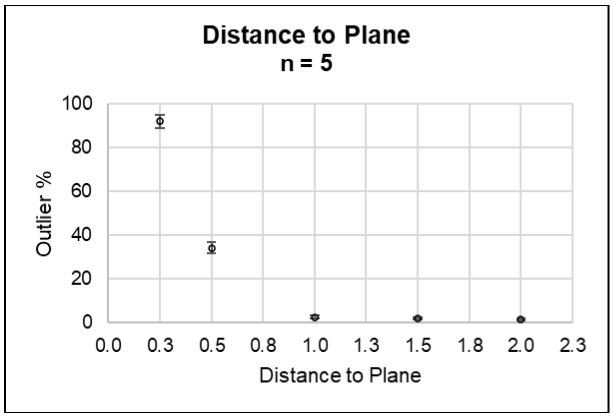
4 Results and Discussion

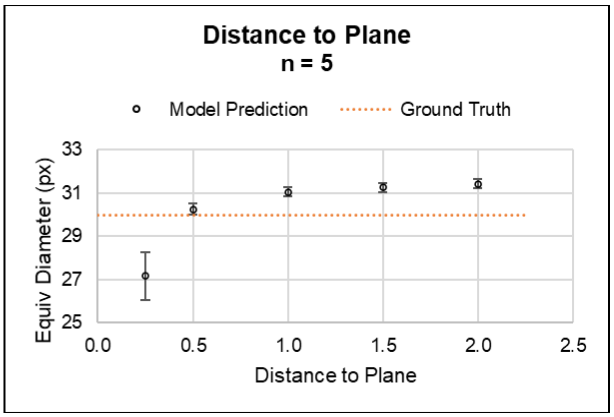
4.1 Sensitivity Analysis on Artificial Network

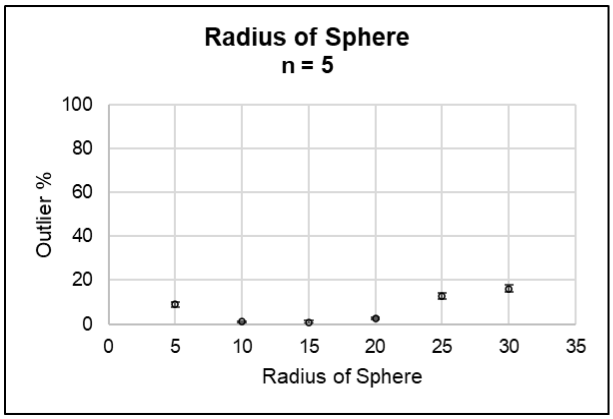
The sensitivity analysis revealed the impact of manipulating *d2plane*, t_c, t_s, σ, and *rMax* on the diameter and outlier fraction output by the model (Figure 11). The baseline hyperparameters used in the sensitivity analysis yielded a diameter of 31.0 ± 0.2 px (n=5), which overestimates the actual diameter (30px) by ~3.5%. The tendency to overestimate could arise from a variety of algorithm-related factors (loss of information through PCA, projecting a volume of coordinates onto a plane, and the conversion of the projection coordinates from Cartesian system to a discrete array representation), or sample-related factors (the artificial network is a discrete representation and so the fiber surface might not necessarily be at distance 15px from the center). Nevertheless, the OPlanaR algorithm appears to be an effective method to measure the diameter of thresholded fibers for many combinations of hyperparameters. Out of the 5 variables tested, the standard deviation of the Gaussian appears to have the least impact on the outlier fraction or measured diameter.

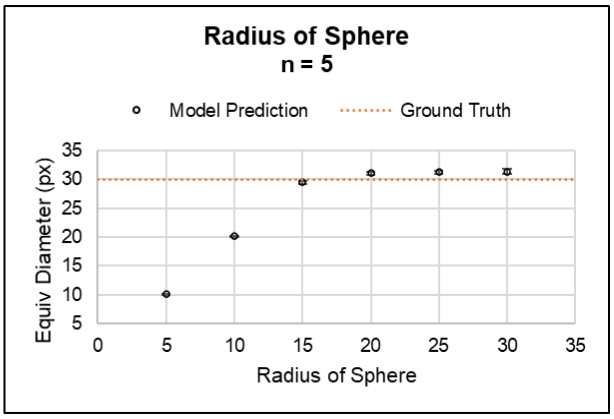
The *d2plane* and *rMax* follow similar relationships versus the measured diameter. As both parameters increase, ceteris paribus, the measured diameter increases until a certain threshold where there is no considerable change to the diameter. As *d2plane* gets smaller, the orthogonal projection gets more prohibitive, resulting in fewer projected points and more irregularly shaped blobs (thus increasing the outlier fraction) with smaller diameters on average. As *rMax* dips below a threshold that represents the theoretical radius of the fiber, the measured diameter decreases proportionately to *rMax*. This is a direct proportionality, where a *rMax* of 5px and 10px yield diameter predictions of ~10px and ~20px, respectively. If *rMax* is smaller than the true radius of a fiber, then only those voxels within the sphere are projected (as opposed to all the points in the fiber). Hence, *rMax* should be set to a value that is higher than the maximum theoretical radius of a fiber in the network to avoid underestimating fiber diameter. Based on the results, *rMax* can be set very high without the measured diameter changing significantly, due to the downstream outlier model.

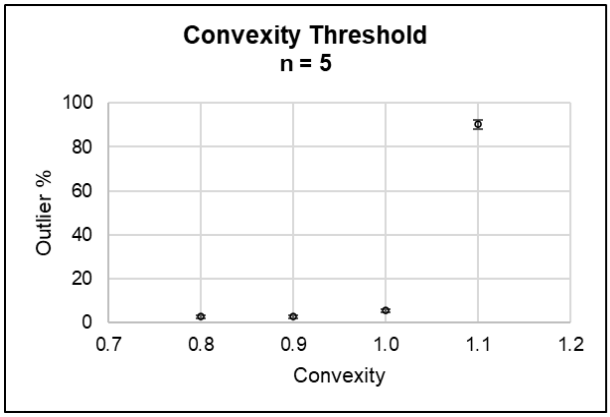
As the solidity and convexity thresholds increase, the outlier fractions increase. However, an increase in outliers flagged by the model does not diminish the diameter measurement significantly, because the outlier model sets stricter conditions on projections (strictly circular), which are ultimately good representations of fiber cross sections.

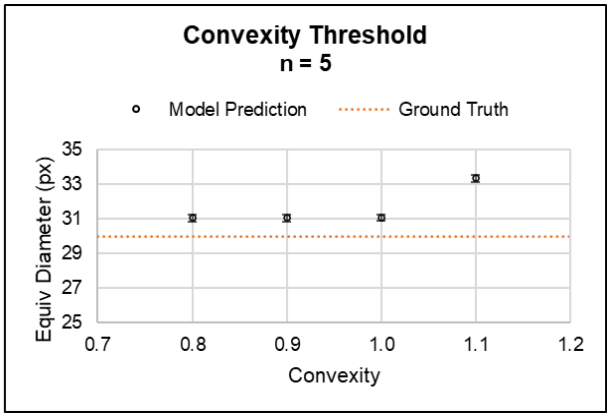

Distance to Plane
n = 5
Outlier %
100
80
60
40
20
0
0.0 0.3 0.5 0.8 1.0 1.3 1.5 1.8 2.0 2.3
Distance to Plane

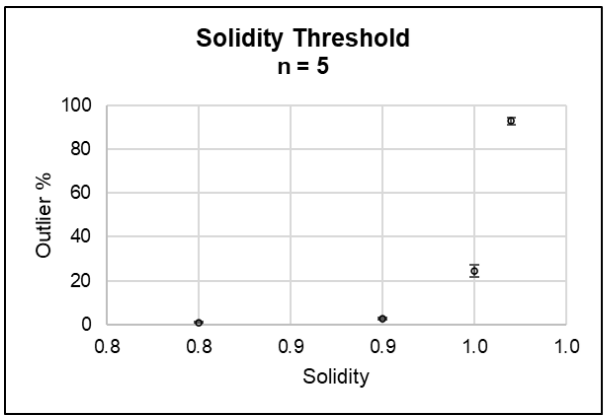

Distance to Plane
n = 5
Model Prediction
Ground Truth
Equiv Diameter (px)
33
31
29
27
25
0.0 0.5 1.0 1.5 2.0 2.5
Distance to Plane

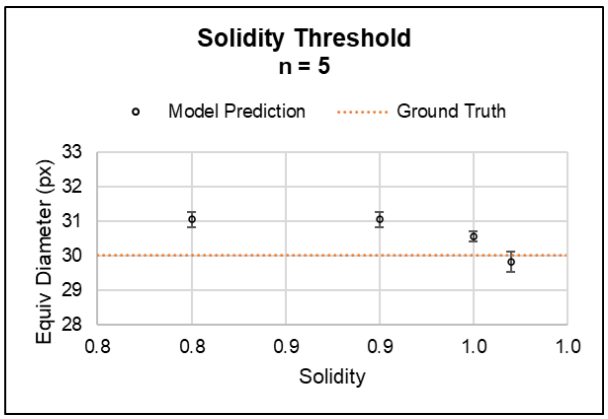

Radius of Sphere
n = 5
Outlier %
100
80
60
40
20
0
0 5 10 15 20 25 30 35
Radius of Sphere


Radius of Sphere
n = 5
Model Prediction
Ground Truth
Equiv Diameter (px)
35
30
25
20
15
10
5
0 5 10 15 20 25 30 35
Radius of Sphere


Convexity Threshold
n = 5
Outlier %
100
80
60
40
20
0
0.7 0.8 0.9 1.0 1.1 1.2
Convexity


Convexity Threshold
n = 5
Model Prediction
Ground Truth
Equiv Diameter (px)
35
33
31
29
27
25
0.7 0.8 0.9 1.0 1.1 1.2
Convexity


Solidity Threshold
n = 5
Outlier %
100
80
60
40
20
0
0.8 0.8 0.9 0.9 1.0 1.0
Solidity


Solidity Threshold
n = 5
Model Prediction
Ground Truth
Equiv Diameter (px)
33
32
31
30
29
28
0.8 0.8 0.9 0.9 1.0 1.0
Solidity

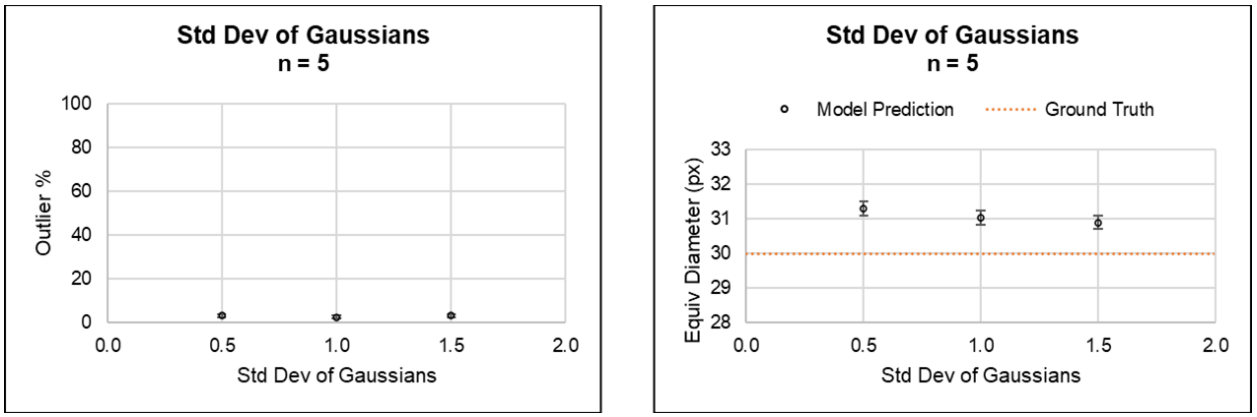


Figure 11: Sensitivity analysis results from experiments on artificial fibers. From top row to bottom row: impact of manipulating *d2plane*, *rMax*, t_c, t_s, and σ on the outlier fraction (left) and measured radius (right). True radius of the cylinder has been included on the graphs on the right. Error bars are based on standard error and the number of samples, n, for the error bars has been included in the title of the graph. Some error bars are very small and so are obscured by the circular marker.

4.2 Model Performance for Different Fiber Densities

The density of the artificial fiber network impacts the OPlanaR algorithm's speed, accuracy, and number of measurements. The error in predicted diameter in Figure 12 was computed using Equation 8. The figure suggests that the diameter predictions increase linearly with image density, causing a linear increase in error from the ground truth. Furthermore, the number of outliers appears to increase linearly with image density before starting to diminish after a certain threshold. The increase in diameter and outliers as density increases could be attributed to the fact that highly clustered networks have large regions of overlapping fibers and yield projections with asymmetric shapes (clustered blobs) and larger sizes. The OPlanaR algorithm's time complexity appears to be of order polynomial time based on image density ($O(n^2)$ where n represents the number of positively thresholded voxels). These experiments highlight the potential accuracy and speed limitations of the OPlanaR algorithm when predicting diameter for dense networks.

Equation 8: Error in diameter calculation

$$Error\ in\ diameter = \frac{Predicted\ diam}{Ground\ truth\ diam} - 1$$

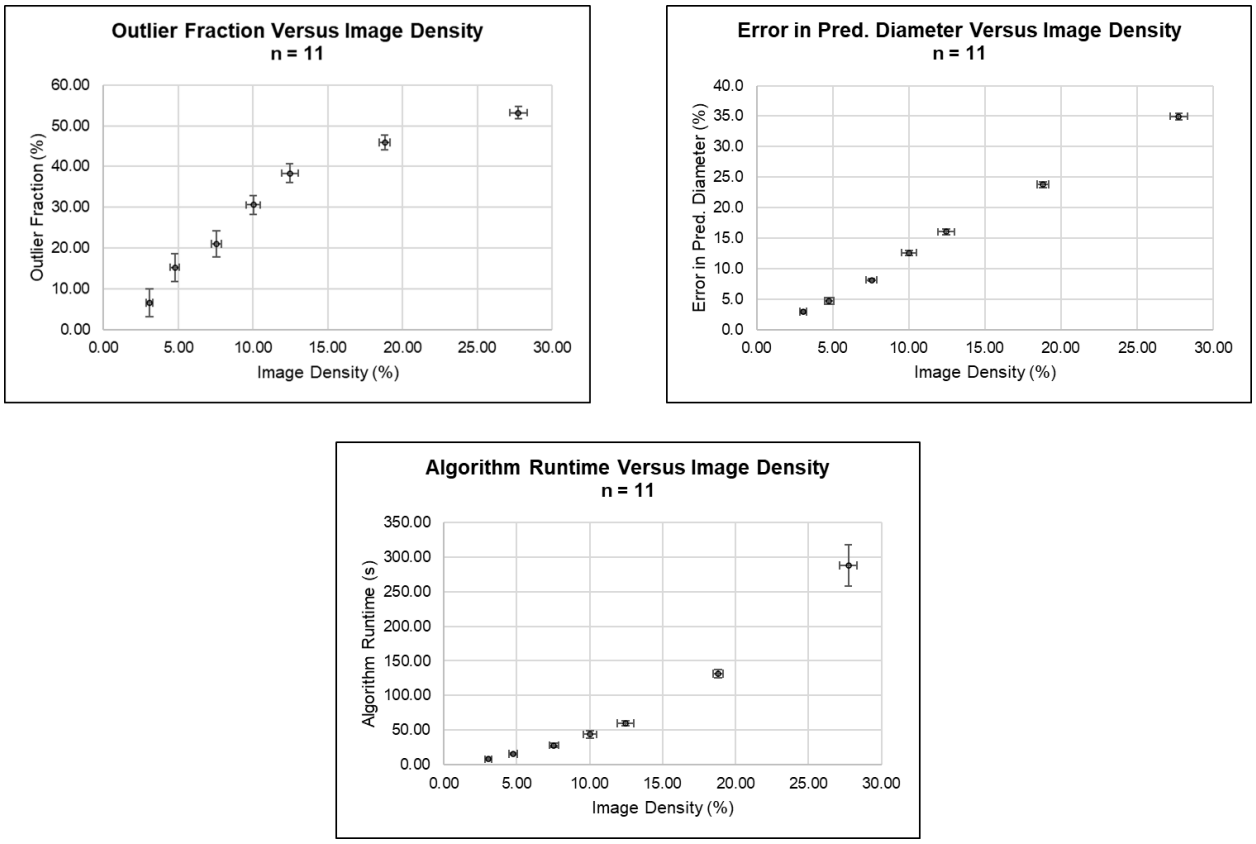


Figure 12: Increasing fiber density on the outlier fraction (top-left), speed (top-right), error in predicted diameter (bottom). Experiments conducted on artificial fiber networks. X and Y Error bars are based on standard error and the number of samples, n, for the error bars has been included in the title of the graph. Some error bars are very small and so are obscured by the circular marker.

4.3 Image Analysis Pipeline Versus Manual Annotation

The experiments on MEndR images demonstrate the feasibility of the image analysis pipeline in measuring equivalent diameter of muscle fibers (Table 1). The model's mean equivalent diameter predictions deviate from the manually annotated diameters by ~6.4% on average considering all 4 images. However, the model and manual annotation distributions do not always agree based on the Kolmogorov-Smirnoff tests (Figure 13), suggesting that the pipeline is not a perfect substitute for manual annotation. The deviations from manual and model diameter distributions could be attributed from the difference in the number of measurements taken by the model (225+) versus the annotator (50). Apart from this discrepancy, the image analysis pipeline seems to be quite quick, taking ~1.7 seconds per diameter measurement. This is approximately 9–17 times faster than a human annotator, assuming a speed of 15–30 seconds per measurement (Figure 14).

It should be noted that these results might not necessarily extend to other MEndR images, because of the small fiber densities considered (~1.14% on average). In addition, it is apparent that

segmentation is the most burdensome stage of the pipeline, taking ~76% of the processing time. Despite these flaws, the pipeline does demonstrate some potential due to its speed and accuracy.

Table 1: Results from applying image analysis pipeline on MEndR images

Image (#)	Image Density (%)	Mean Diameter (microns)		Dev. from Manual (%)
		Model Prediction	Manual Annotation	
1	1.34	8.7 ± 0.2	8.6 ± 0.3	1.2
2	0.97	6.4 ± 0.2	7.4 ± 0.4	13.5
3	1.33	7.4 ± 0.2	8.2 ± 0.3	9.8
4	0.95	8.5 ± 0.2	8.6 ± 0.3	1.2

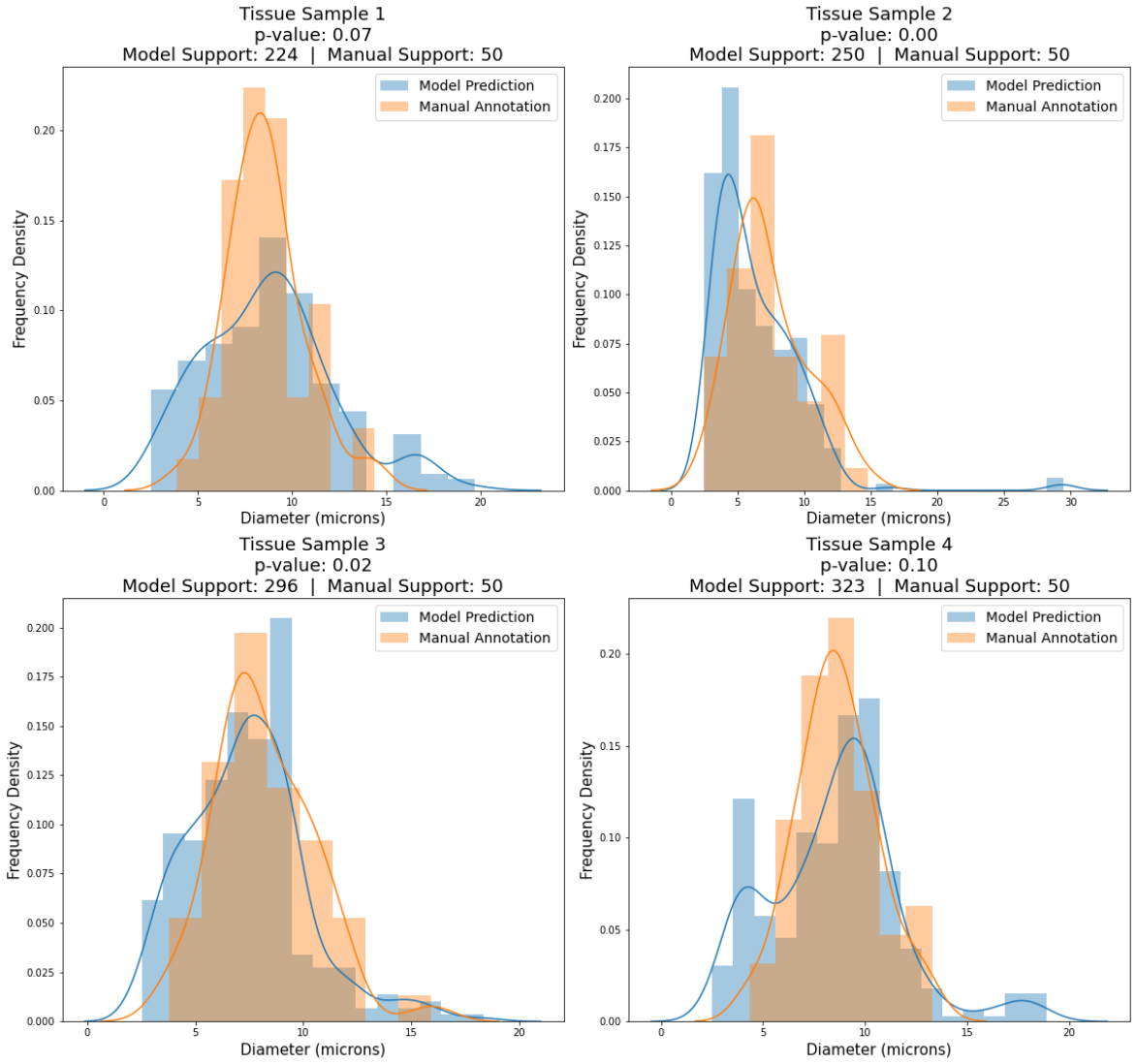


Figure 13: Distribution plots of manual annotation overlayed over model prediction for equivalent diameter for 4 MEndR images. P-values for the 2-sided KS test have been included, along with the number of measurements taken by the model and the manual annotator. The null hypothesis states that the two distributions are similar because their cumulative distribution functions are similar. Assuming a cutoff p-value of 0.05, we should reject the null hypothesis for tissue samples 2 (top right) and 3 (bottom left) and accept the null hypothesis for tissue samples 1 (top left) and 4 (bottom right).

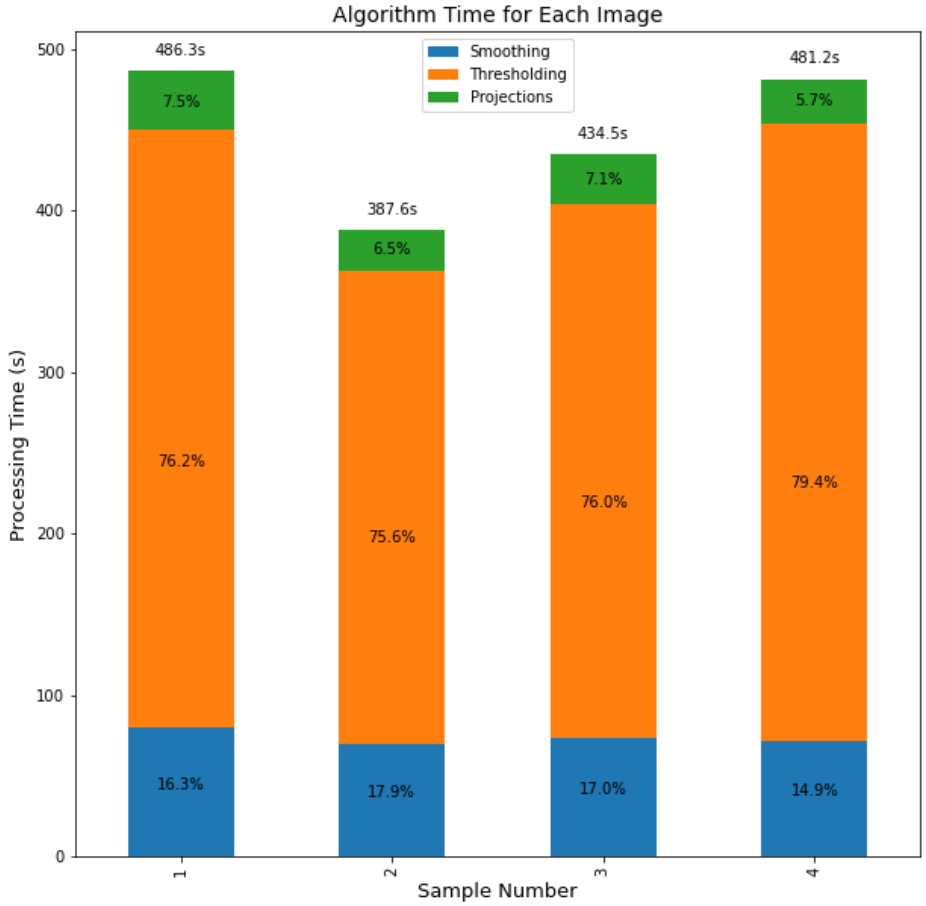


Figure 14: The time taken for each stage of the image analysis pipeline when applied to MEndR images. Smoothing includes all pre-processing steps of min-max normalization and anisotropic diffusion smoothing. The thresholding involves the modified graph cuts segmentation. Projections refers to the application of the OPlanaR algorithm.

4.4 Limitations & Future Work

There are several limitations associated with the experimental and algorithmic approaches in this project. The sensitivity analysis and fiber density experiments (described in Section 3.1.3) revealed that the OPlanaR algorithm typically overestimates diameter; a phenomenon that exacerbates as fiber density increases. These algorithmic issues did not seem to interfere significantly with the experiments on MEndR images – particularly when comparing mean diameter – because the images considered for the project were very low density. Nevertheless, future work should be dedicated to the development of methods that can quantify the errors in diameter for different fiber densities using artificial networks. An accurate quantification of the systematic error associated with the model at different fiber densities could be used to recalibrate diameter measurements calculated on real MEndR images.

Another improvement that could help quantify errors would be to create artificial networks that more accurately simulate real MEndR images. The artificial network in the experiments described in Section 3.1.3 consisted of cylindrical rods with straight spines and identical radii. In reality, one fiber in a MEndR image typically has a distribution of diameter measurements and a non-linear spine. Hence, there is a lot of room to create more accurate simulations of fiber networks with non-linear spines and asymmetric diameters. The development of better simulated networks could also help quantify the error of the OPlanaR algorithm in networks with increased asymmetry. It should be noted that the average diameter in asymmetric networks should be computed based on the length of the fibers (as opposed to averaging over the number of rods).

An algorithmic approach to address the overestimation of fiber diameter would be the implementation of a more sophisticated outlier detection system. The outlier detection system only accounted for two features (convexity and solidity) and used simple hard constraints for classification. Future researchers should consider using more shape or morphological features in the outlier system. A richer feature set would not only allow for additional constraints on the projections (ellipticity could control how elliptical projections could look, for example), but also enable the application of machine learning-based algorithms including Random Forest classifiers or Support Vector Machines that could result in more robust outlier detection systems.

A more important long-term goal for this project could involve the implementation of a "human-in-the-loop" Graphical User Interface (GUI) to enhance model interpretability and usability (Figure 15). An interpretable model would allow researchers to apply the logic and mathematics of the algorithm without any expertise in coding, spurring increased adoption. For example, a front-end visualization for the OPlanaR algorithm might contain two panes: (i) one that includes a view of the 3D binarized image with the vertex in the skeleton highlighted, and (ii) one that displays the orthogonal projection for that vertex. MEndR researchers could then tag the orthogonal projections as outliers/non-outliers using the front-end interface. A machine learning-based model could then be trained or re-trained on the backend based on the technician's outlier annotations. Finally, the trained machine learning model can be used as the outlier detection system for the OPlanaR algorithm. The front-end would be useful in visually demonstrating how the model projects fibers at unique vertex locations in the fiber skeleton and how these projections can look irregular in areas of high fiber overlap. This system is especially useful because a researcher does not need any programming expertise to run the model.

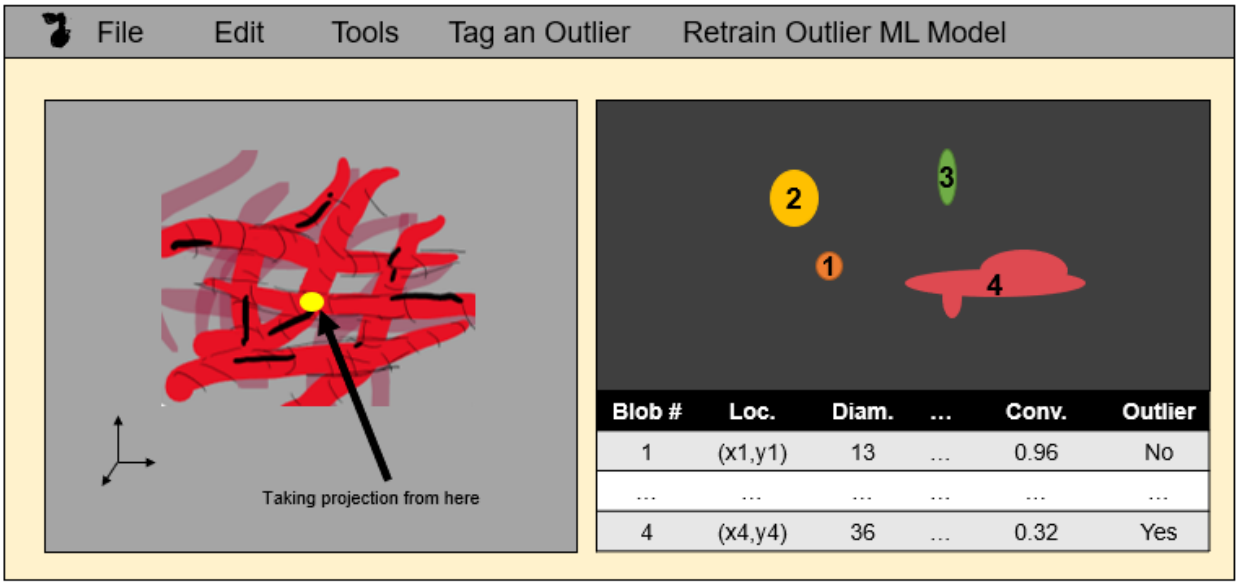


Figure 15: Front-end GUI that can help improve model interpretability and usability. The left pane is a 3D rendering of the muscle tissue with the vertex of the orthogonal projection highlighted. The right pane visualizes the projections and tabularizes information related to the blobs (features and whether the model classified the blob as an outlier). Researchers have the option to override the predictions of the machine learning-based outlier model with their own annotations and retrain the ML model with the new annotations using buttons in the toolbar at the top.

Given the scope of the project, little effort was invested in rigorously improving, testing, and validating each step of the model with real MEndR images. For instance, there was no validation done to prove that graph-cuts was the most optimal segmentation technique – the approach was chosen because it had been applied in previous literature on 2D images. Furthermore, there was no parameter tuning conducted for the graph-cuts technique; this report used same parameters as Guo et al [3]. Experiments should be conducted that compare graph-cuts to alternative classical methods like Otsu's method, or more sophisticated methods like U-Nets.

In addition, the speed of the pipeline could be improved upon. For example, the graph-cuts segmentation was incredibly slow (Figure 14). This could be, in part, attributed to the fact that the image was split into 16 blocks and operated on sequentially due to computational memory constraints. Hence, one might consider exploring the use of map-reduce or parallel computation schemes available through distributed computing platforms (Hadoop, PySpark) or deep learning libraries (TensorFlow, PyTorch) to improve the speed of the segmentation. Apart from segmentation, the OPlanaR algorithm's execution speeds were far slower for images with high density compared to those of low density (Figure 12). To address the polynomial time of the algorithm, future researchers might consider subsampling the thresholded points in the fiber network (as opposed to projecting all the voxels in the image).

Finally, there was little effort dedicated to the validation of the skeletonization algorithm. A fundamental assumption of the work is that the skeleton passes through the medial axis of the fiber. The project made use of MATLAB functions for this operation, which were not necessarily tailored to highly clustered fiber networks. In fact, in some images, it was observed that the skeletonized fiber occasionally did not pass through the medial axis of the fiber, especially if that fiber was in an area of high density. An improperly constructed skeleton interferes with the orthogonal projection and can generate more than one blob, impacting the model's accuracy. Hence, it would be worthwhile to validate the performance of skeletonization for dense networks, and investigate the use of alternative skeletonization schemes designed for fiber networks [9].

5 Conclusion

This report set out to design an algorithm that could extract the distribution of muscle fiber diameters from 3D images produced by the MEndR model, to address limitations in the speed of analysis. To achieve its aims, the study first presents an orthogonal projection and outlier detection algorithm (OPlanaR) that can measure the diameter of thresholded fibers. The OPlanaR algorithm performed reasonably well when tested on artificially generated fiber networks, with an error of ~3.5% against the true diameter for a baseline set of hyperparameters. However, the OPlanaR algorithm appeared to perform worse in both speed and accuracy as images increased in density. This work then implemented the OPlanaR algorithm as the processing stage within an image analysis pipeline, applying this pipeline to 4 MEndR images with manually annotated diameters. The pipeline appears to perform well when comparing the mean diameters of the model with those produced by manual annotation; observing a ~6.4% deviation on average between the two. However, these results are not conclusive because the images tested were very low density and the statistical distributions of manual annotations and model predictions did not agree for half of the images. To address these limitations, future work should be dedicated to quantifying the systematic errors associated with dense fiber networks, validating different stages of the pipeline, and improving the outlier detection system using statistical learning algorithms. Nevertheless, this work demonstrates the potential of the OPlanaR algorithm and the image analysis pipeline in obtaining measurements of the diameter of fibers from 3D images, and thus, addressing bottlenecks in MEndR analysis.

6 Bibliography

[1] E. A. Tasnadi, T. Toth, M. Kovacs, A. Diosdi, F. Pampaloni, J. Molnar, F. Piccinini and P. Horvath, "3D-Cell-Annotator: an open-source active surface tool for single-cell segmentation in 3D microscopy images," *Bioinformatics*, vol. 36, no. 9, pp. 2948-2949, 2020.

[2] N. Carragher, F. Piccinini, A. Tesei, O. J. Trask Jr., M. Bickle and P. Horvath, "Concerns, challenges and promises of high-content analysis of 3D cellular models," *Nature Reviews Drug Discovery*, vol. 17, no. 8, pp. 606-606, 2018.

[3] Y. Guo, X. Xu, Y. Wang, Z. Yang, Y. Wang and S. Xia, "A computational approach to detect and segment cytoplasm in muscle fiber images," *Microscopy research and technique*, vol. 78, no. 6, pp. 508-518, 2015.

[4] T. Falk, D. Mai, R. Bensch, O. Çiçek, A. Abdulkadir, Y. Marrakchi, A. Böhm, J. Deubner, Z. Jäckel, K. Seiwald, A. Dovzhenko, O. Tietz, C. D. Bosco, S. Walsh, D. Saltukoglu, T. L. Tay, M. Prinz, K. Palme, M. Simons, I. Diester, T. Brox and O. Ronneberger, "U-Net: deep learning for cell counting, detection, and morphometry," *Nature Methods*, vol. 16, pp. 67-70, 2019.

[5] T. Hu, Q. Xu, W. Lv and Q. Liu, "Touching Soma Segmentation Based on the Rayburst Sampling Algorithm," *Neuroinformatics*, vol. 15, no. 4, pp. 383-393, 2017.

[6] A. R. M. Fard, "Development of a 96-well Platform to Increase Throughput of a Muscle Endogenous Repair Model," University of Toronto, Toronto, 2020.

[7] C. H. Comin, X. Xu, Y. Wang, L. Costa and Z. Yang, "An image processing approach to analyze morphological features of microscopic images of muscle fibers," *Computerized medical imaging and graphics*, vol. 38, no. 8, pp. 803-814, 2014.

[8] M. Miazaki, M. Viana, Z. Yang, C. H. Comin, Y. Wang, L. F. Da Costa and X. Xu, "Automated high-content morphological analysis of muscle fiber histology," *Computers in Biology and Medicine,* vol. 63, pp. 28-35, 2015.

[9] A. M. Stein, D. A. Vader, L. M. Jawerth, S. A. Weitz and L. M. Sander, "An algorithm for extracting the network geometry of three-dimensional collagen gels," *Journal of Microscopy,* vol. 232, no. 3, pp. 463-475, 2008.

[10] J. Jiang, P. Kao, S. A. Belteton, D. B. Szymanski and B. S. Manjunath, "Accurate 3D Cell Segmentation using Deep Feature and CRF Refinement," 2019.

[11] J. Jonkman, C. M. Brown, G. D. Wright, K. I. Anderson and A. J. North, "Tutorial: guidance for quantitative confocal microscopy," *Nature Protocols,* vol. 15, pp. 1585-1611, 2020.

[12] Y. Xu, T. Wu, F. Gao, J. R. Charlton and K. M. Bennett, "Improved small blob detection in 3D images using jointly constrained deep learning and Hessian analysis," *Scientific reports,* vol. 10, no. 1, pp. 1-12, 2020.

[13] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on pattern analysis and machine intelligence,* vol. 12, no. 7, pp. 629-639, 1990.

[14] S. Sternberg, "Biomedical Image Processing," *Computer*, vol. 16, no. 1, pp. 22-34, 1983.

[15] J. Chen, L. Ding, M. P. Viana, M. C. Hendershott, R. Yang, I. A. Mueller and S. M. Rafelski, "The Allen Cell Structure Segmenter: a new open source toolkit for segmenting 3D intracellular structures in fluorescence microscopy images," bioRxiv, 2018.

[16] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 9, no. 1, pp. 62-66, 1979.

[17] Y. Guo, X. Xu, Y. Wang, S. Xia and Z. Yang, "An image processing pipeline to detect and segment nuclei in muscle fiber microscopic images," *Microscopy research and technique,* vol. 77, no. 8, pp. 547-559, 2014.

[18] O. Daněk, P. Matula, C. Ortiz-de-Solórzano, A. Muñoz-Barrutia, M. Maška and M. Kozubek, "Segmentation of Touching Cell Nuclei Using a Two-Stage Graph Cut Model," in *Proceedings of the 16th Scandinavian Conference on Image Analysis*, 2009.

[19] I. Arganda-Carreras, V. Kaynig, C. Rueden, K. W. Eliceiri, J. Schindelin, A. Cardona and S. H. Seung, "Trainable Weka Segmentation: a machine learning tool for microscopy pixel classification," *Bioinformatics*, vol. 33, no. 15, pp. 2424-2426, 2017.

[20] C. Sommer, C. Strähle, U. Köthe and F. A. Hamprecht, "ilastik: Interactive Learning and Segmentation Toolkit," in *2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, Chicago, 2011.

[21] O. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox and O. Ronneberger, "3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation," in *International conference on medical image computing and computer-assisted intervention*, 2016.

[22] D. Ciresan, A. Giusti, L. M. Gambardella and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in *Advances in neural information processing systems*, 2843-2851, 2012.

[23] J. Chen, L. Yang, Y. Zhang, M. Alber and D. Z. Chen, "Combining fully convolutional and recurrent neural networks for 3d biomedical image segmentation," in *Advances in neural information processing systems*, 2016.

[24] Y. Boykov and G. Funka-Lea, "Graph cuts and efficient ND image segmentation," *International journal of computer vision*, vol. 70, no. 2, pp. 109-131, 2006.

[25] Y. Boykov and V. Kolomogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 9, pp. 1124-1137, 2004.

[26] J. Chen, S. Banerjee, A. Grama, W. J. Scheirer and D. Z. Chen, "Neuron Segmentation Using Deep Complete Bipartite Networks," in *MICCAI 2017*, 2017.

[27] M. R. Jeong, B. C. Ko and J. Y. Nam, "Overlapping nuclei segmentation based on Bayesian networks and stepwise merging strategy," *Journal of Microscopy,* vol. 235, no. 2, pp. 188-198, 2009.

[28] Y. Chen, H. Sun, H. Yang and X. Pan, "Level set method of cell image segmentation based on combinations of edge, region and prior information," in *2017 4th International Conference on Systems and Informatics (ICSAI)*, Hangzhou, 2017.

[29] H. Su, F. Xing, J. D. Lee, C. A. Peterson and L. Yang, "Automatic Myonuclear Detection in Isolated Single Muscle Fibers Using Robust Ellipse Fitting and Sparse Representation," *IEEE/ACM transactions on computational biology and bioinformatics,* vol. 11, no. 4, pp. 714-726, 2013.

[30] J. Chen, Q. Dou, L. Yu and P. Heng, "VoxResNet: Deep Voxelwise Residual Networks for Volumetric Brain Segmentation," arXiv, 2016.

[31] T. Janssens, L. Antanas, S. Derde, I. Vanhorebeek, G. Van den Berghe and F. G. Grandas, "CHARISMA: An integrated approach to automatic H&E-stained skeletal muscle cell segmentation using supervised learning and novel robust clump splitting," *Medical image analysis,* vol. 17, no. 8, pp. 1206-1219, 2013.

[32] J. Mula, J. D. Lee, F. Liu, L. Yang and C. A. Peterson, "Automated image analysis of skeletal muscle fiber cross-sectional area," *Journal of Applied Physiology,* vol. 114, pp. 148-155, 2013.

[33] F. Liu, A. L. Mackey, R. Srikuea, K. A. Esser and L. Yang, "Automated image segmentation of haematoxylin and eosin stained skeletal muscle cross-sections," *Microscopy,* vol. 252, no. 3, pp. 275-285, 2013.

[34] The MathWorks, *MATLAB Image Processing Toolbox R2019a,* Natick, Massachusetts: MATLAB, 2019.

[35] P. Kollmannsberger, M. Kerschnitzki, F. Repp, W. Wagermaier, R. Weinkamer and P. Fratzl, "The small world of osteocytes: connectomics of the lacuno-canalicular network in bone," *New Journal of Physics,* vol. 19, no. 073019, 2017.

[36] D. Young, "Gradients with Gaussian smoothing," MATLAB Central File Exchange, 2020.

[37] M. A. Wirth, "Shape Analysis & Measurement," Image Processing Group, Guelph, 2004.

[38] D. Lopes, "Anisotropic Diffusion (Perona & Malik)," MATLAB Central File Exchange, 2007.

[39] R. Gauriau, "Shape-based approaches for fast multi-organ localization and segmentation in 3D medical images," Télécom Paris, Paris, 2015.

[40] S. Eddins, "Image Graphs," MATLAB File Exchange, 2020.

[41] M. Miazaki, M. P. Viana, Z. Yang, C. H. Comin, Y. Wang, L. F. Da Costa and X. Xu, "Automated high-content morphological analysis of muscle fiber histology," *Computers in biology and medicine,* vol. 63, pp. 28-35, 2015.

[42] M. Miazaki, M. P. Viana, Z. Yang, C. H. Comin, Y. Wang, L. F. Da Costa and X. Xu, "Automated high-content morphological analysis of muscle fiber histology," *Computers in biology and medicine,* vol. 63, pp. 28-35, 2015.

7 Appendices

7.1 Appendix A

The source code for the work that went into this project can be found in the *src/code* folder. The final code written in MATLAB can be found in the *final_matlab* folder. The code for the preliminary Python work can be found in the *preliminary_python* folder. Each of these folders contains a README in markdown format which describes how to set up and run the code on a Windows computer.

7.2 Appendix B

This appendix describes the equations used for the Gaussian smoothing filter and the centered differences filter. The equations used to create the Gaussian smoothing filter have been included in Equation 9 and Equation 10.

Equation 9: Function to compute the size of the 1D Gaussian filter

$$size = 2 * \lceil 2.6\sigma \rceil + 1$$

Equation 10: Function to compute the value of the filter at a position x in the array. The center of the array is assumed to be at x=0. Positions to the left of the center decrease by 1 and positions to the right of the center increase by 1

$$k_\sigma(x) = \frac{1}{\sigma(2\pi)^{\frac{1}{2}}} \cdot \exp\left(-\frac{x^2}{2\sigma^2}\right)$$

Setting a σ of 1 produces a filter of size [7,1]. (x=0 at the center of the array. x increments by 1 as you move into positions that are to the right of the center. x decrements by 1 as you move to positions to the left of the center). The output of the filter in this case would be [0.0044, 0.0540, 0.2420, 0.3989, 0.2420, 0.0540, 0.0044].

The centered differences filter to compute the vectors tangent to the skeleton have been included in Equation 11.

Equation 11: Centered difference filter used to compute the tangent vectors

$$f_g = [-0.5, 0, 0.5]$$

7.3 Appendix C

The sensitivity analysis tested the impact of 5 out of the 6 hyperparameters of the OPlanaR model and assumed a base set of hyperparameters (Table 2). Each hyperparameter was changed from the base set in Table 2 and the resulting performance was measured. The set of hyperparameters tested has been included in Table 3.

Table 2: Base set of hyperparameters used in the sensitivity analysis

Hyperparameter	Base Value
d2plane	1
t_c	0.90
t_s	0.90
σ	1
rMax	20
lMin	10

Table 3: The set of hyperparameters tested as part of the sensitivity analysis

Hyperparameter	Base Value
d2plane	{0.25, 0.50, 1, 1.5, 2}
t_c*	{0.8, 0.9, 1, 1.1}
t_s	{0.8, 0.9, 0.95, 0.97}
σ	{0.5, 1, 1.5}
rMax	{5, 10, 15, 20, 25}

Note: The convexity threshold should theoretically not exceed 1. However, since convexity was computed using user defined functions (as opposed to MATLAB's *regionprops* function), the convexity did go above 1 for many blobs. This could be, in part, due to rounding errors and the inconsistency of the user-defined function with MATLAB's *regionprops* function