Individual fiber segmentation of three-dimensional microtomograms of paper and fiber-reinforced composite materials

Jens Bache-Wiig Per Christian Henden

29th July 2005

Preface

This paper is a Master's thesis in Computer Science. It was written by two Master students at the Department of Computer and Information Science at the Norwegian University of Science and Technology (NTNU). The thesis was written at the Paper and Fiber Research Institute (PFI). The duration of the work on the Master's thesis at NTNU is 22 weeks. This thesis had a slightly longer duration due to other scientific work simultaneously performed by the authors.

The project was supervised by Prof. Richard Blake from NTNU and Dr. Ing. Per Nygård from PFI.

Intended audience and how to read this thesis

It has been assumed that the reader has a basic knowledge of computer science and mathematics. Concepts in paper science, physics, image analysis and graphics are explained only where the understanding of these concepts are necessary to understand other parts of the report. Concepts that are less central are not explained, but the reader is given references that enables him or her to learn these concepts from other sources. We believe this explanation of common concepts to be necessary for the report to be read by people from all fields interested in the analysis of high-resolution microscopy images. This includes material science, biology, chemistry, computer science and others.

Acknowledgments

The authors would like to thank Richard Blake and Per Nygård for their suggestions and feedback on the thesis. Thanks goes to Sabine Rolland du Roscoat and Xavier Thibault at the European Synchrotron Radiation Facility (ESRF) for their assistance in the lab at ESRF and for answering our questions about the image acquisition process. Special thanks go to Jarle Anfinsen and Jan Roger Henden for proof reading a draft of this thesis.

Contact information

Per Christian Henden	perchrh@pvv.org	`http://www.pvv.org/~perchrh/`
Jens Bache-Wiig	jensbw@gmail.com	`http://www.mrpixel.tk`

Abstract

The structure of a material is of special significance to its properties, and material structure has been an active area of research. In order to analyze the structure based on digital microcopy images of the material, noise reduction and binarization of these images are necessary. Measurements on fiber networks, found in paper and wood fiber – reinforced composites, require a segmentation of the imaged material sample into individual fibers.

The acquisition process for modern X-ray absorption mode micro-tomographic images is described. An improved method for the binarization of paper and fiber-reinforced composite volumes is suggested. State of the art techniques for individual fiber segmentation are examined and an improved method is suggested.

Software tools for the mentioned image processing tasks have been created and made available to the public. The orientation distribution of selected paper and composite samples was measured using these tools.

Contents

Chapter 1

Introduction

This chapter presents the background of this thesis. In addition, the structure of the thesis is explained, other related ongoing projects are mentioned, and limitations of this work are pointed out.

1.1 Background

Recent developments in microscopy and computing power have made digital measurements on high-resolution images of fibrous materials possible. These measurements are useful for scientists wishing to gain a better understanding of the structure and other properties of the material at micro level. Digital measurements afford automation of measurement procedures and thereby objective results. In order to perform digital measurements on the images each point in the image must be recognized as material and non-material. For some measures it is necessary to partition the material further, e.g. into individual fibers. The separation of different parts of the image is commonly called segmentation.

The Norwegian Paper and Fiber institute (PFI) in cooperation with the Norwegian University of Science and Technology (NTNU) has previously worked on digital measurements on paper. This thesis can be viewed as a continuation of the thesis by Marit Hagen and Runar Holen last year (2004) (Holen and Hagen 2004). According to Rune Holmstad, a PHD-student that did his doctoral thesis for PFI, it it still an open question whether a perfect segmentation of the fiber network is possible (Holmstad 2004). PFI is interested in new tools and methods that will help them in the characterization of paper and new wood-based composite materials. Measurements of three-dimensional fiber alignment is of special interest.

As students of computer science and image processing we see the work on analyzing these images as an interesting challenge.

The images we will be working on were acquired at the European Synchrotron Radiation Facility (ESRF) through a co-operative project named "Structure Characterization of Wood Fiber-based Materials", spring 2005.

1.1.1 Structure of the thesis

The first chapter gives an introduction to the report, and its background. Chapter two explains the method of image acquisition used and the contents of the images we have worked on. The third chapter is a review of relevant literature. Readers with a background in image processing might want to skip this chapter. Chapter four explains our method of separating bulk fiber mass from other parts of the volumes. The fifth chapter explains our method of separating individual fibers from each other. Chapter six chapter gives the results of measurements on the processed images. The seventh chapter gives an overview of the implementation of the mentioned methods. The eight and final chapter is a discussion of our findings and ideas for future work.

1.2 Other ongoing efforts

This section gives a list of other people working on image processing tools for the segmentation of paper fibers in x-ray micro tomography volumes from ESRF known to us.

Maria Axelsson[1] is working at Centrum för bildanalys, Uppsala, Sweden as a doctoral student. Her goal is to do a true 3D segmentation of the fiber volume.

Tuomas Turpeinen[2], a researcher at the department of physics, the university of Jyväskylä, Finland, is working on reducing the ring artifacts in the volumes, and will later work on segmentation of the volumes.

Sabine Rolland du Roscoat[3], a doctoral student at ESRF, is working on segmentation of paper. She has completed a method for segmenting the volumes into fibers and fibrils, fillers and background and plans to continue with looking at the segmentation of individual fibers.

[1] `http://www.cb.uu.se/~maria/`
[2] `http://www.phys.jyu.fi/homepages/turpeinen/index.html`
[3] `http://www.esrf.fr`

1.3 Limitations

The amount of available computer memory is a limiting factor for the segmentation procedure. As a result of this we have mainly worked on volumes cropped to 25% of their original size, and in 8-bit gray-level resolution instead of 16 or 32-bit. The initial segmentation (Chapter 4) is best using 16-bit data, as the noise suppressing filter does not work with 32bit floating point data. This is an implementation issue and can be fixed. The cropping of image data is not believed to have any ill effects. The fiber segmentation assumes that all fibers have well-defined lumens.

Chapter 2

Image processing problem

We will be working on high-resolution 3D images of paper as well as images of wood fiber-reinforced composite materials in an epoxy-vinyl ester matrix, which is a type of plastic. These images were obtained at ESRF. The image acquisition method is explained in Section 2.1.

Our task is to first separate each of the images by its phases, that is void and fiber for the paper images, and plastic and fiber for the composites. We term this the *initial segmentation*. The next step is to separate the mass of fibers into individual fibers. We term this the *individual segmentation*. The segmentations are necessary in order to do measurements on the images.

The statistical properties of the images are presented in Section 2.2, and domain knowledge of wood-fibers in Section 2.3.

We shall perform different measurements on the segmented volumes. Accuracy is important when measuring, and as such, both segmentations should be good approximations to the physically correct[1] segmentations.

The last section, Section 2.4 deals with measurements on images of paper and fiber-reinforced composites.

2.1 Image acquisition

The properties of an image depend on the method by which it was acquired. This section explains how our image data was acquired.

A verbose explanation like this one is not found elsewhere in the literature, to our knowledge. We believe this explanation to be important for the

[1]Note that our digital images are only samples of the true, continuous materials, and, strictly speaking, any measurements on them will be approximations because of this.

understanding of the images we will investigate in the rest of this paper, and we have certainly missed such an explanation ourselves.

The preferred technique for acquiring three-dimensional high-resolution images of paper is X-ray micro tomography (Holmstad 2004), as this method gives access to the paper structure non-invasively at a uniform high resolution and contrast in all spatial directions and all parts of the imaged volume.

The main disadvantages associated with X-ray micro tomography are its high cost and low availability. Desktop-sized equipment exists, but currently does not match the contrast of large synchrotron sources. A synchrotron is a large, circular particle accelerator. We have used the accelerator at ESRF, which has a diameter of 143 meters, and produces images with resolution in the range of $0.3\mu m$ to $30\mu m$. In more experimental setups, resolution down to $0.05\mu m$ is possible and there are plans achieve even higher resolution.

Skyscan, a manufacturer of room- and desktop-sized equipment, reports that their Skyscan-2011 NanoTomograph detects details down to $0.15\mu m$ (SkyScan 2005). Future developments are likely to make high resolution X-ray micro tomography more affordable and available (Holmstad 2004).

The rest of this section describes the current[2] micro tomography method of image acquisition at beamline ID19 at ESRF. The description is based on a visit to the facility by the authors, and the article Rolland du Roscoat et al. (2005).

Preparations

Samples are prepared by first cutting out a small rectangular material sample of maximum size $2\times2mm$. Then a post-it of roughly the same dimensions are cut out. The post-it is glued, with its glue-side pointing upwards, to the end of a small glass rod (tube). When the glue is dry, the sample is attached to the post-it on top of the tube (figure 2.1). This avoids the problem of having glue seep into the sample, that complicated the interpretation of previous images, where the sample was glued directly to the tube. The tubes used are of glass and have a filled core. Previously, capillaries with a hollow core were used. These capillaries were highly visible in the image compared to the new ones, which are easily removed. In addition they were more difficult to attach the sample to.

The capillary is mounted between the detector, which is a special type CCD [3], and the X-ray beam (figure 2.2). The sample is then moved electronically

[2] as per April 2005

[3] A charge-coupled device (CCD) is a sensor for recording images, consisting of an

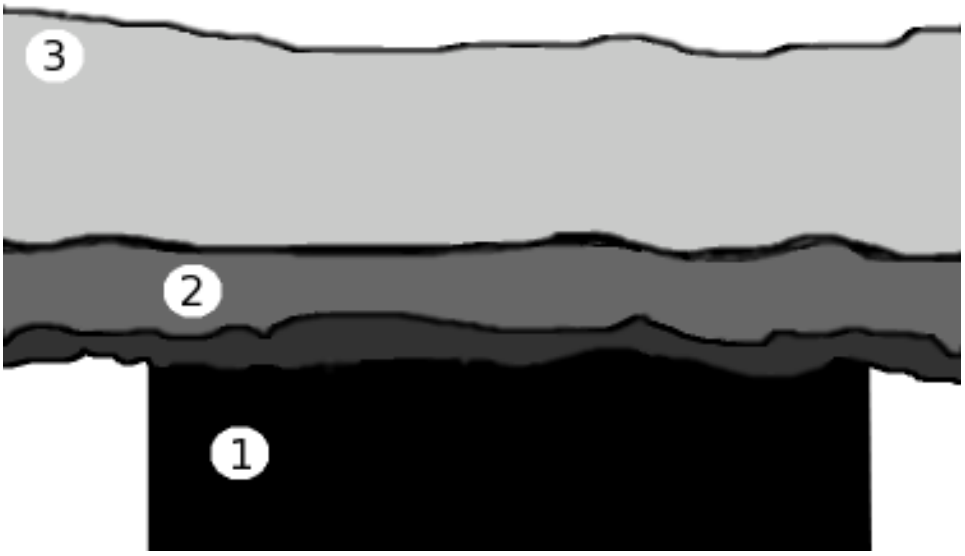


Figure 2.1: Sample mounting sketch. 1: Glass tube, 2: Post-it, 3: Paper. Between layers there is glue.

to select a region of interest of $1.4mm \times 1.4mm$, parallel to the detector. The size of the image in the direction perpendicular to the detector (height of sample) depends on the placement of mechanical components called *slits*. There are several slits. They are closed to avoid diffusion and to increase the quality of images in terms of signal to noise ratio, and opened to increase the field of view.

The camera time (ct) is set. The camera time is the length of beam time per projection. The acquisition process lasts slightly longer than the number of projections times ct seconds. The camera time affects the contrast of the resulting image, with optimal ct depending on the sample and the status of the synchrotron. A ct value of 0.4 seconds was used for all of our samples. This was considered a good all-round value by ESRF staff.

Method

A beam of high-energy photons is directed at the sample which is to be imaged (figure 2.2). A custom-built CCD (Labiche et al. 1996) behind the sample records an image. The sample is then rotated and a new image is recorded. These projection images are called radiographs. Some 1500 radiographs are taken while rotating the sample 180°. These radiographs, divided

integrated circuit containing an array of linked, or coupled, capacitors. Under the control of an external circuit, each capacitor can transfer its electric charge to one of its neighbors (Wikipedia 2005).

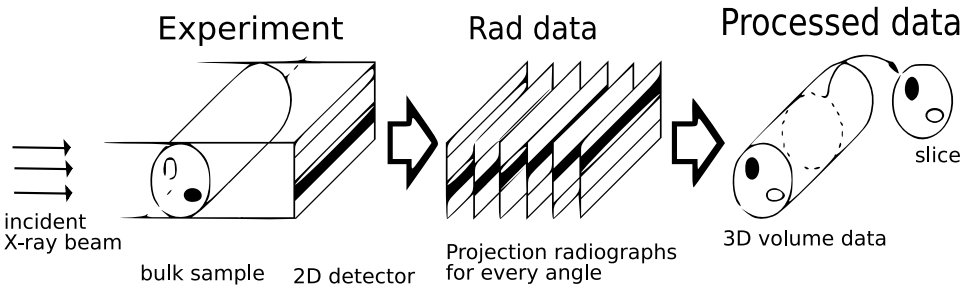


Figure 2.2: Experimental setup

point wise with corresponding reference images, are used to reconstruct the 3D volume. The reconstruction is done using a custom version of the filtered back-propagation algorithm and results in a cylinder-shaped volume image. The CCD has a depth of 2^{14} bits.

Noise reduction is performed on the resulting images. The noise is assumed to be independent and additive at each point in the CCD's matrix. The noise is reduced by taking 20 images of the void, capturing only noise, finding the median image of these 20 images, and subtracting the median image from the sample images.

As paper is sensitive to humidity and temperature conditions, its structure may evolve during the experiments (Rolland du Roscoat et al. 2005). Consequently, after the scan, two more radiographs of the sample at 0°and 90°are recorded. By computing the cross-correlation between the measured radiographs during and after the scan, the swelling of the sample during the data acquisition is evaluated. A correction is performed, resulting in higher contrast and better defined shapes.

The distance (d) from CCD to sample is of some importance. Images taken at a value of $d \simeq 0$ are called absorption mode images. Images taken at larger distances (commonly in the range 50–100 mm) are called phase-contrast images. The difference between the two is that while the former records average atomic density of the material, the latter record borders between regions of different refractive indexes. The value 0.0 in an absorption mode image means that the volume in that location contains mostly air or void.

Some materials, like paper-epoxy vinyl composites, should be imaged using a d of 50–100 mm, while paper benefits from using the smallest d possible. This is due to the different diffusion of X-Ray beams from the materials.

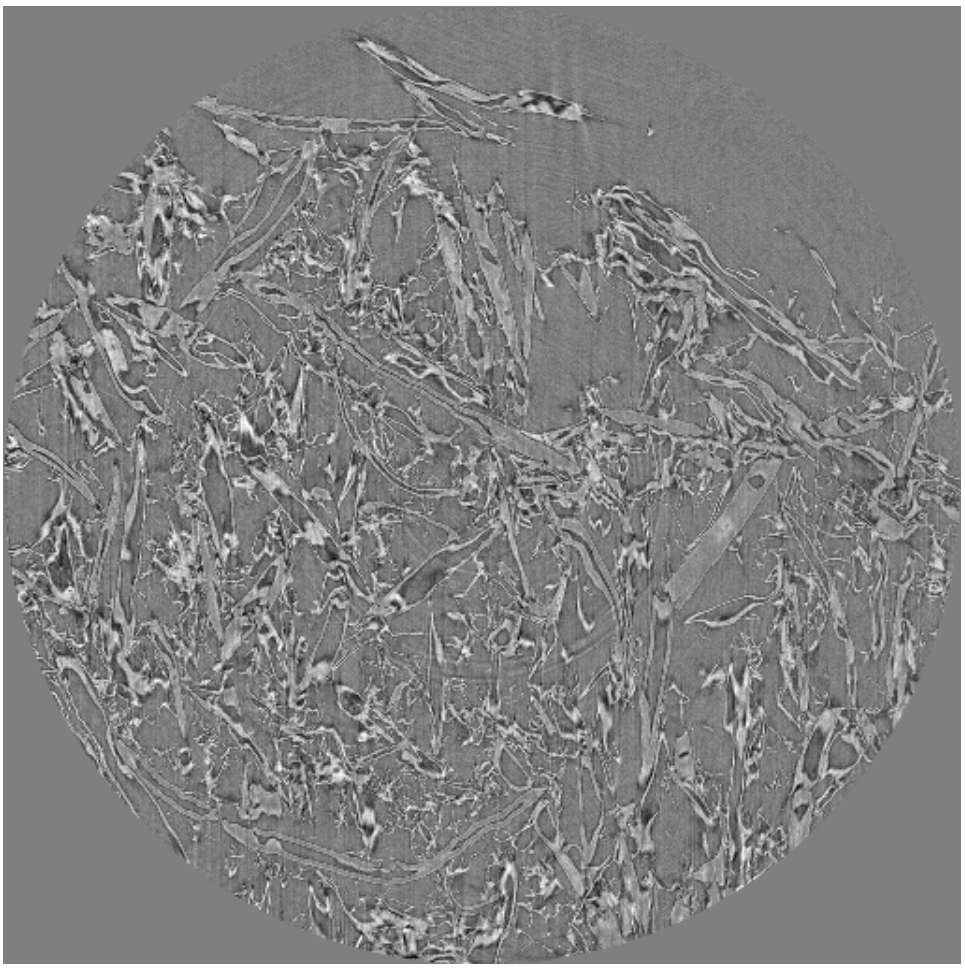


Figure 2.3: A sample paper-volume cross-section

Result

The procedure results in a volume-image of a physical area of $1.4mm^2 \times$ height, giving 3D volumes in 32-bit float format (figure 2.3). Height is restricted by the slit positions, which decides the maximum field of view, or some lower, manually chosen value[4].

Optionally, the volume can be converted to 8-bit unsigned integer format on-site. This is often necessary because of the sheer size of the 32-bit volumes. The size of the 32-bit volumes range from 4.4GB to 35GB in $0.7\mu m$ resolution.

The conversion is done by first selecting a range given by a minimum and a maximum value. The range is divided into 256 equally sized bins, and values from the 32-bit float image are mapped linearly into these bins, resulting in an 8-bit image.

The choice of this range is not straightforward. The extremal values of the volume are not necessarily equal to the extremal values of the volume of interest, as the image often shows more than the material to be examined later, e.g. air. In addition the images contain noise.

A human observer must inspect the 32-bit volume and decide on the range by looking at the histogram of regions that she identifies as the material of interest. Possibly by guessing at the intensity distribution of the material in the image, based on peaks in the histogram, and choosing the minimum and maximum from this distribution.

2.2 Statistical properties of the volume

We will look at images of two main types of materials in this paper. This is images of paper and images of fiber-reinforced composites (figure 2.4). We will use the volumes S8a (paper) and STFI7 (fiber-reinforced composite) as examples.

The images show the material of interest (the foreground), other materials (the background) and noise of different types. The histogram of a paper sample commonly shows two peaks, one peak corresponding to the mean of the image background distribution, and one corresponding to the mean of the paper distribution. An example is shown in figure 2.5(a). We shall label this example volume 'S8a'. The second peak in this image is a little to the right of the tallest peak. The two distributions overlap, even after noise reduction.

[4] The volumes are cropped when imaging paper, because paper usually is quite thin ($0.2 - 0.4$mm high).

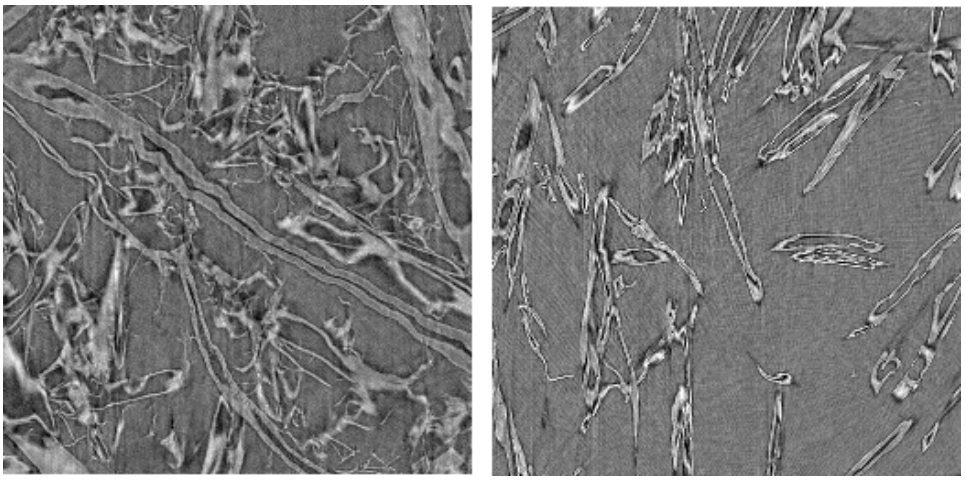


Figure 2.4: Sample of volume S8a, TMP-based newsprint paper (left) and sample of volume STFI7, wood fibers in epoxy vinyl ester, a type of plastic

The background distribution can be examined by looking at the pixel intensities in parts of the volume where there is no material present. It is easily sampled by deliberately sampling void when acquiring the image.

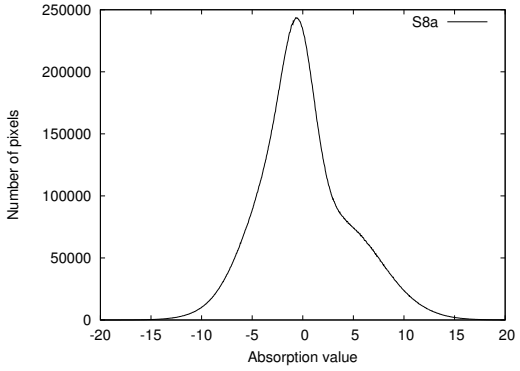
The background distribution (with noise) of S8a is shown in figure 2.5(b).

The volumes suffer from a special type of noise. Ring-shaped artifacts appear in the images (figure 2.6). They overlap in intensities with both background and foreground. The amount and characteristics of the ring artifacts depend on multiple devices used for the setup (multilayer, CCD camera, scintillator defects), and can be magnified by the sample itself.

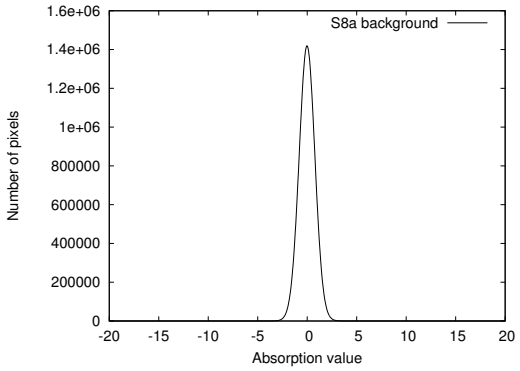
The circles are actually semi-circles, arcs with length 40-60% of the circle perimeter. When the volumes are rotated, these semi-circles show themselves to be parts of hollow semi-tori in 3D (figure 2.7).

An approximation to the distribution of pixel intensity values of the material of interest can be calculated by binarizing the volume into foreground and background and masking the volume with the binary volume, so that the pixel sites belonging to the foreground keep their value and others are set to 0. The foreground distribution of S8a, obtained in this way, is shown in figure 2.5(c). The binarization procedure is explained in Chapter 5. Note that this is an approximation and will make underestimates on the left side of the mean, due to the binarization method used.

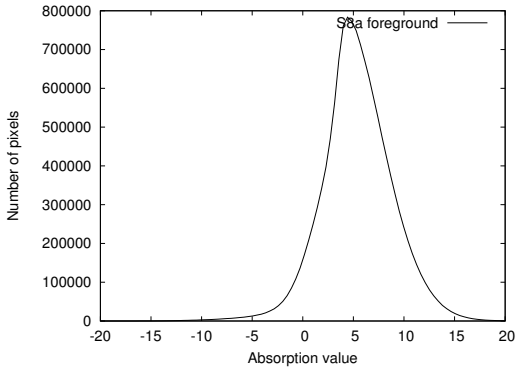
There is less contrast between background and fibers in STFI-7 than in normal paper volumes (figure 2.8). This applies to all the epoxy vinyl ester composites we have examined, and we believe it to be caused by more overlap between the two distributions epoxy vinyl ester and wood fibers than air and



(a) Complete



(b) Background only



(c) Foreground (approximated)

Figure 2.5: Histograms of S8a

Figure 2.6: Sample of S8a-background, center of image

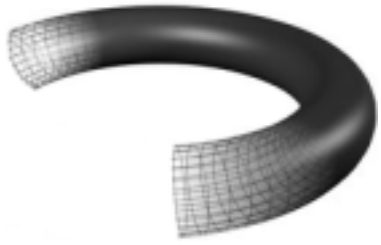
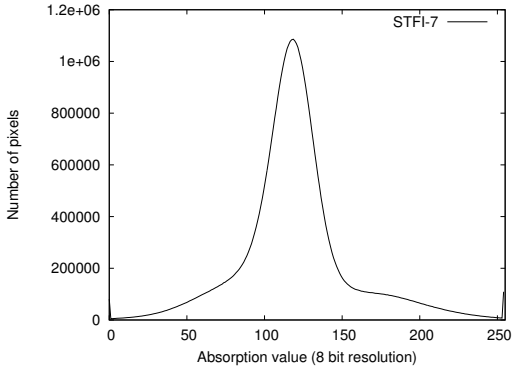
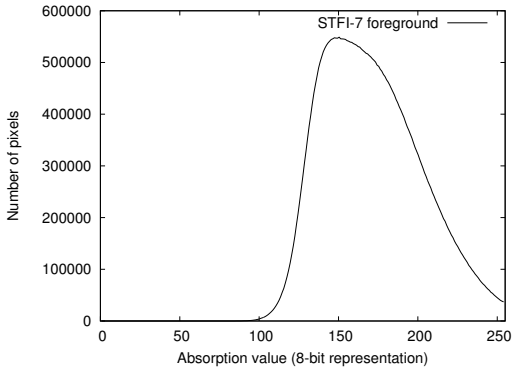


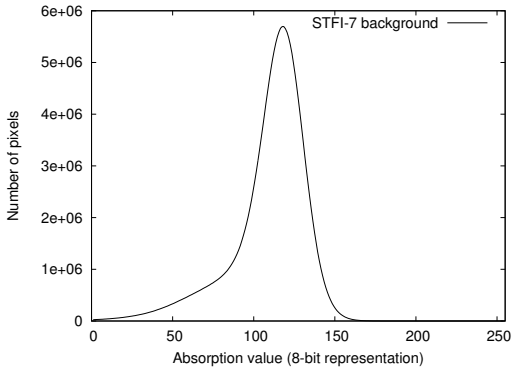
Figure 2.7: Illustration of a hollow semi-torus



(a) Complete



(b) Foreground



(c) Background

Figure 2.8: Histograms of STFI-7 (approximated)

wood fibers. Indeed, when doing an approximation of the foreground and background distributions of STFI-7 as with S8a, and using the inverse mask to get the background, the histograms show that the range of overlap is about 60 8-bit units, which in 32-bit corresponds to about 14.4 units (the range -29.2–32.9 was linearly mapped into the range 0–255), while the overlap in S8a is about 5.0 32-bit units.

The fibers in the composites have roughly the same distribution as in the normal paper volumes, with a peak at about 4.0 and similar shapes. They are a bit difficult to compare because the 8-bit conversion distorts the histogram. The background distribution is different, since it is no longer mostly air, but epoxy vinyl ester, a solid. It ranges from (in 32-bit) approximately -29.2 to 9.2, in contrast to about -2.5 to 2.5 in the paper case. Interestingly, there is considerably less ringing in the composites we have examined than in the paper volumes. The paper volumes and composites were acquired at about the same time with the same experimental setup, save that the composites required a larger slit size and larger distance from the CCD in order to be imaged.

2.3 Domain knowledge

This section explains the high-level properties of the previously mentioned image types.

Paper fibers are hollow. The cavities are called lumina or lumens (singular form: lumen). Lumens are shaped roughly like ellipses in 2D cross-sections, and like long, twisted tubes in 3D. A fiber's lumen is surrounded by its fiber wall. Fibers can collapse (figure 2.9, from Fellers and Norman (1998)) because of external forces exerted on the fiber. Fiber walls have small holes for transport of water and other molecules from one fiber to another. Additionally, fiber walls can crack (figure 2.10) and fragment (figure 2.11). Note also how the fiber shape changes from slice to slice, and how fiber shape varies. Thin-walled fibers often have larger and more cracks than thick-walled ones (Holen and Hagen 2004).

The fiber structure is highly interconnected (figure 2.12), with many contact areas. Indeed, a test[5] showed that more than 90% of the fibers in one volume[6] touched each other either directly or via a path through other fibers.

[5] A 6-connected 3D flood fill on a binarized volume

[6] PH15, a volume image consisting of short chemical fibers and polymer

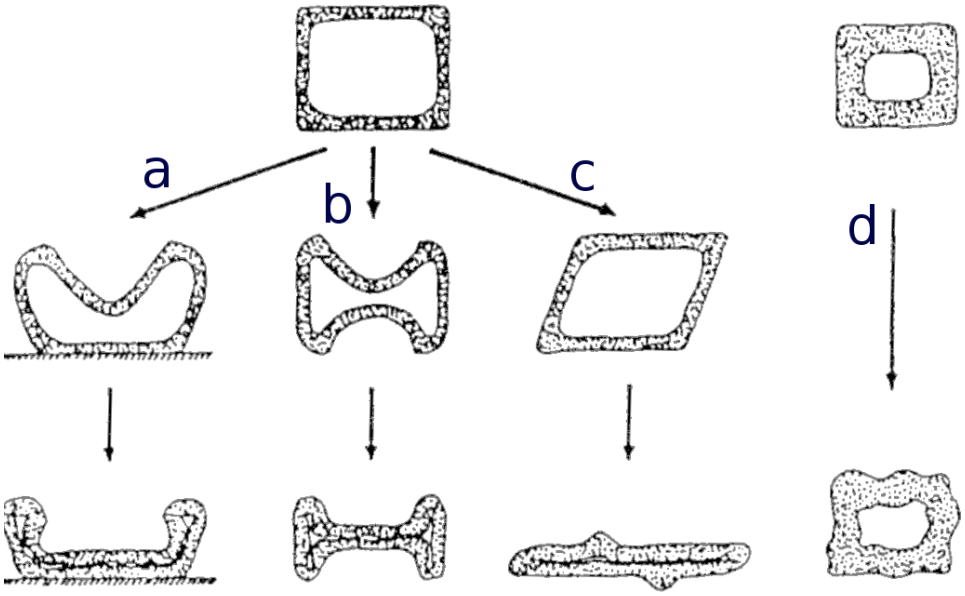


Figure 2.9: Types of collapsed fibers for band fibers (left) and tube fibers (right)

Figure 2.10: Example of cracks in a fiber's wall

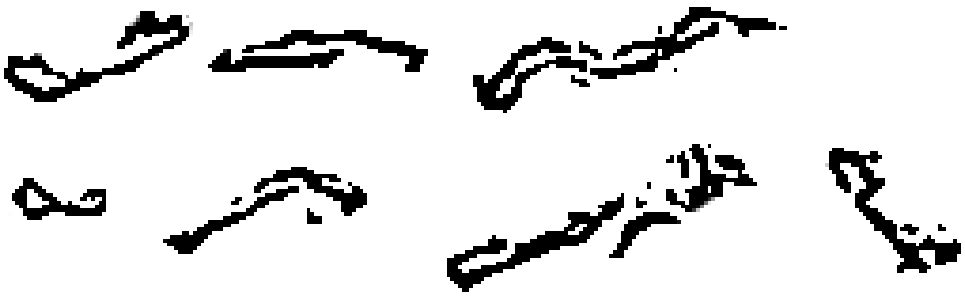


Figure 2.11: Examples of fragmented fibers

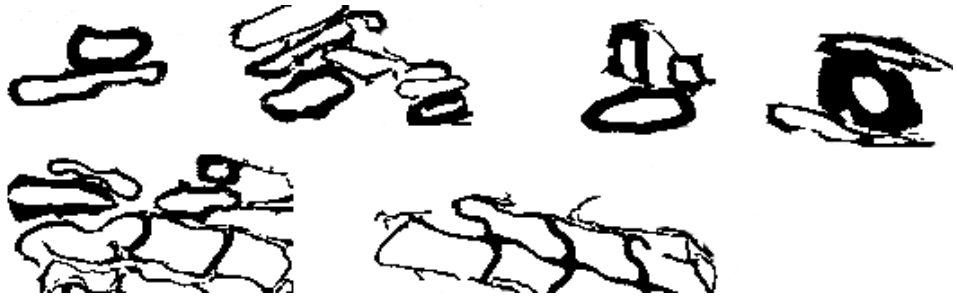


Figure 2.12: Examples of touching borders

2.4 Measurements

There are in principle three levels, or groups, of measurements (Aronsson 2002, p. 47) on images of fibers. Each have different requirements and produce estimates on different scales. They are the paper level, the fiber network level and the fiber level.

On the *paper level* measurements do not require individually labeled fibers, only a binarized digital 3D image of a paper sample. Much research has gone into quantifying the structural properties of paper and other porous materials. To use the same terminology on both paper and fiber-reinforced composite materials, we will use the term *bulk level* instead of paper level. We define the bulk level to be the binarized fiber structure of paper and fiber-reinforced composites, the same as the paper level is for paper. The measurements can be further divided into four types. A list of implemented measurements known to the authors is given for each type.

Surface properties surface area (Holmstad 2004) and surface roughness (Wang et al. 2004)

Material properties thickness, layer density, basis weight and material orientation distribution (Holmstad 2004)

Pore properties porosity and pore chord distribution (Holmstad 2004), pore hydraulic radius distribution (Ramaswamy et al. 2001) and pore size distribution (Huang et al. 2002)

Transport properties diffusion tortuosity, flow permeability and flow tortuosity (Holmstad 2004) and conductivity (Arns et al. 2001)

For measurements at the *fiber network* level, individually labeled fibers are required. The number of contact points between fibers, average free fiber length and average contact area was measured in Aronsson (2002). A method for measuring bonded fiber area was developed in Holen and Hagen (2004).

Finally, we have the *fiber level*, where measurements are done on each fiber, disregarding information about the other fibers. Statistics are used to summarize these measurements. Statistics on fiber length, wall thickness, curvature, twist, kink, torsional rigidity, aspect ratio, degree of collapse, lumen volume, fiber wall volume and bending resistance was measured in Aronsson (2002). The fiber orientation distribution, average curl, average fiber length and average fiber crossing angle was measured in Yang (2001).

An overview of methods for measurements on fibrous structures can be found in Bache-Wiig and Henden (2004).

2.5 Discussion

The cutting and mounting of samples is done manually. It is the authors' experience that it is difficult to cut that small samples manually, and especially difficult to align them on the capillary so that the sample is level with the capillary's surface. It would certainly be helpful if samples could be cut and mounted by a machine or with the help of more advanced tools than scissors.

We found that what seemed like good ranges for 8-bit conversion in the hustle and bustle of the on-site laboratory were not that good when examined further at a later time. Based on this we strongly recommend saving both the 32-bit and 8-bit volumes, so that it can be done if the need to convert the volume to 8-bit using different extrema values should arise.

If it proves too expensive or impractical to store 32-bit versions of the composites (requires 35GB at $0.7\mu m$ resolution), a sub-volume should be stored, so that at least the statistics of the volume can be examined using this sample.

As a final note on selecting the range for 8-bit conversion; it seems that for paper volumes and some composites the values below 0.0 is not of interest as they do not occur in the fibers. These values are best left out (i.e. set to 0) when converting to an 8-bit representation.

The distributions of air (background) and paper (foreground) overlap in the paper volumes, as well as matrix (background) and wood-fiber (foreground) in the composite volumes. We believe this overlap to be caused by the presence of materials with approximately the same average atomic number in background and foreground. Epoxy vinyl ester and wood fibers both contain much hydrogen, carbon and oxygen, while air contains mostly oxygen and nitrogen. This can explain the overlap between epoxy vinyl ester and wood fibers, and the smaller overlap between air and wood fibers, and implies

that doing noise reduction on 32-bit data instead of 8 or 16-bit data will not result in two separated distributions.

Chapter 3

Review of relevant literature

This chapter explains image-processing terms we will be using in later chapters.

3.1 Shape descriptors

In this section we will look at methods for describing regions found in digital images (shapes) by shape descriptors. A shape descriptor is a measure done on a region. It is a statistic, as it describes the set of points that makes up the shape by a vector or scalar.

Area. The *area* of a shape can be found by counting pixels belonging to that shape. If a value in meters is needed, this area is multiplied with the area of one pixel. For example, 1000 pixels in $0.7\mu m^2$ resolution is $0.49mm^2$.

Circularity. The *circularity* (compactness) of a shape is the square of its perimeter divided by its area. It can be normalized to be 1.0 for circles by dividing with 4π.

$$\text{circularity} = \frac{P^2}{4\pi A} \tag{3.1}$$

Statistical moments. The statistical central moment $m_{p,q}$ of a binary shape is given by (3.2), where $\bar{x}$ means the average of coordinate x of the shape.

$$m_{p,q} = \sum (x - \bar{x})^p (y - \bar{y})^q \tag{3.2}$$

The area of a shape is $m_{0,0}$. The average in the x-direction, $\bar{x}$ is $\frac{m_{1,0}}{m_{0,0}}$, and in the y-direction, $\bar{y}$ is $\frac{m_{0,1}}{m_{0,0}}$. The centroid, i.e. coordinates of the mass-center, is $(\bar{x}, \bar{y})$.

Eccentricity. The *eccentricity* (elongation) is the ratio of the maximum length of line or chord that spans the region to the minimum length chord.

These chords can be difficult to measure. An approximation can be found by using statistical moments to calculate the eccentricity ϵ (Marshall 1997).

$$\epsilon = \frac{m_{2,0} + m_{0,2} + \sqrt{(m_{2,0} - m_{0,2})^2 + 4(m_{1,1})^2}}{m_{2,0} + m_{0,2} - \sqrt{(m_{2,0} - m_{0,2})^2 + 4(m_{1,1})^2}} \tag{3.3}$$

Curvature. The *curvature* of the border of a shape can be used as a descriptor of that shape. The border is a planar curve.

For a plane curve C, the curvature at a given point P has a magnitude equal to the reciprocal of the radius of a circle that closely touches the curve at P (figure 3.1, from Wikipedia (2005)). For example, a circle of radius r will have curvature $\frac{1}{r}$ (Weisstein 2005).

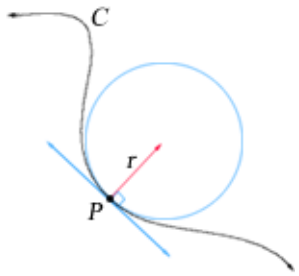


Figure 3.1: Illustration of the curvature concept

For a plane curve given parametrically as $c(t) = (x(t,), y(t))$, the curvature is given by (3.4) (Mackworth and Mokhtarian 1988).

$$\kappa = \frac{\dot{x}\ddot{y} - \dot{y}\ddot{y}}{(\dot{x}^2 + \dot{y}^2)^{3/2}} \tag{3.4}$$

The estimation of curvature is highly sensitive to noise, because of the second-derivatives in the formula (Flynn and Jain 1989). It is therefore of interest to smooth the curve before estimating curvature. One way of doing this is by curve evolution (Mackworth and Mokhtarian 1988), where the curve is convolved with a differentiated Gaussian kernel to smooth

it and differentiate it at the same time. $x(t)$ and $y(t)$ are convolved independently of each other with the same kernel. This is valid because $D(x(u) * g(u,\sigma),u) = x(u) * D(g(u,\sigma),u)$, where $D(f,x)$ is the differentiation operator, and $*$ is the convolution operator. Both the single and double derivatives can be obtained this way.

Bending energy. The *bending energy* B of a shape with N contour points P is given by equation (3.5) (Young et al. 1974).

$$B = \frac{\sum_{p \in P} \kappa(p)^2}{N} \tag{3.5}$$

The bending energy quantifies the energy stored in the shape of the contour.

Convex area. *Convex area* of a shape is the smallest area that is both convex and include the complete shape (Russ 1995). In other terms, it is the area of the convex hull of the shape. A shape S is said to be convex if the straight line segment joining any two points in S lies entirely in S. The convex hull H of S is the smallest convex set containing S (Gonzalez and Woods 2001).

There are many ways to calculate the convex area of a shape. One of them is the gift wrapping algorithm (Wikipedia 2005):

> The gift wrapping algorithm begins with a point A known to be on the convex hull, e.g., the leftmost point, and selects the point B on the set border such that all points are to the right of the line AB. This point may be found on O(N) time by comparing polar angles of all points with respect to point A taken for the center of polar coordinates. Repeating with B and so on until one reaches A again yields the convex hull in K steps. The gift wrapping algorithm is exactly analogous to the process of winding a string (or wrapping paper) around the set of points.

The convex area can also be found by using mathematical morphology (Gonzalez and Woods 2001, p.539).

Solidity. We define *solidity* is the ratio of area to convex area.

Convexity. We define *convexity* is the ratio of convex area to convex perimeter squared.

3.2 The medial axis transform

The *medial axis transform*, the result of which is often referred to as a *skeleton* (figure 3.2), refers to a simplified representation of the original three-dimensional data where important information about the topology and geometry of the original structure is maintained.

The medial axis can be defined by the concept of maximal circles, or spheres (3D). If we sum up all the centers of all maximal circles entirely contained within the structure, we get a thinned result consisting only of lines and surfaces (Sonka et al. 1999).

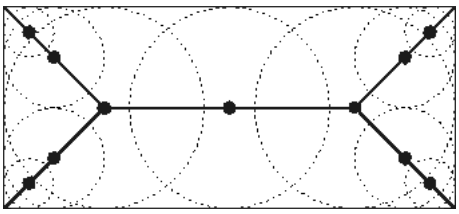


Figure 3.2: The skeleton of a rectangle

3.3 Mathematical morphology

Mathematical morphology can be viewed as a mathematical way of specifying transformations of images. The result of a morphological transform on an image is always a new image. Key elements of morphology is to think of the image as a topographical surface (for gray-level images) or as plain sets (for binary images), and to define transforms by one or more masks, called structural elements, that are moved across the image.

3.3.1 Dilation and erosion

The dilation and erosion transforms are the basic transforms in mathematical morphology. They are explained below. All transforms in mathematical morphology can be expressed using erosions and dilations.

Let the set of points in the image that is the image foreground (object) be known as A. Let the structuring element be known as B. The eroded A is the union of all coordinates where B can be translated to and be completely contained within A. The erosion of A with B can then be expressed as: $A \ominus B = \{a | B_a \subseteq A\}$. Erosion can for example be used to extract inner borders, using borders $= A - (A \ominus B)$.

Similarly, the dilated A is the union of all coordinates where B can be translated to and intersect with A. The dilation of A with B can then be expressed as $A \oplus B = \{a | B_a \bigcap A \neq \emptyset\}$. Dilation can for example be used to extract outer borders, using borders = $(A \oplus B) - A$.

Dilation and erosion, as well as a number of other morphological transforms can be extended to apply to gray-scale images as well (Sollie 2003).

3.3.2 The hit-or-miss transform (HMT)

HMT uses two structural elements. One of them must match (hit) the background and the other match the foreground in order for a pixel to be changed by the transform. Using the terminology from the previous section, the HMT can expressed mathematically as $A \circledast B = (A \ominus B_1) \bigcap (A^c \ominus B_2)$, where B_i is structural element i, and A^c is the complement of the set A, meaning in this case the background of image A. HMT can for example be used to remove isolated pixels, i.e. object pixels that are surrounded by background pixels, e.g. a single white dot surrounded by black dots.

3.3.3 Chamfer distance transforms

A distance transform takes a binary image as input and calculates for each pixel in the background, the distance to the foreground (or vice verse). A measure of distance must be decided on. We will use Euclidean distance, i.e., the distance between two points is the length of the shortest line between them.

Distance transforms are not exclusive to mathematical morphology, but the transforms used in later chapters will all be based on morphology.

A naive algorithm for computing the distance transform is to for each background pixel, calculate the distance to all foreground pixels, and store the minimum. This algorithm uses $O(nN)$ operations, where n is the number of object pixels in the image, and N is the number of background pixels. A quicker method is to propagate local integer distances. This can be viewed as an approximation to the exact distances. These types of algorithms use $O(nm)$ operations, where m is the local area size.

An algorithm for finding the most efficient algorithms for the exact, euclidean, distance transform in $O(n)$ operations is presented in Shih and Wu (2004), but as it fails to obtain correct results in some special cases (Shih and Wu 2004), we will not discuss it further. Instead, we will focus on the morphology approach, which is explained below.

First pass:
For i = 2,3, ..., lines do
  For j = 2,3, ..., columns do
    For k = 2,3, ..., layers do
      $v^1_{i,j,k} = \min_{(p,q,r)\in\mathcal{N}}\{V^0_{p,q,r} + d_n\}$

Second pass:
For i = lines-1,..., 2, 1 do
  For j = columns-1,..., 2, 1 do
    For k = layers-1,..., 2, 1 do
      $v^2_{i,j,k} = \min_{(p,q,r)\in\mathcal{N}}\{V^1_{p,q,r} + d_n\}$

Table 3.1: Two-scan distance transform algorithm. v^0 is the initial image, $\mathcal{N}$ is the set of already visited neighbors in the scan, and d_n is the appropriate local distance

The input image is transformed by two structural elements, called the "forward" and the "backward" mask. The elements in these masks are different weights (local distances) a, ..., e for different directions (table 3.3).

The algorithm by Borgefors (1996) (table 3.1) takes an image as input, where all background points have value 0 and all object points have value 1. Distance to background is computed in the following manner: For all pixels scanned from start of the image to the end, for all positions in the forward mask, centered in the current location, find the minimum sum of mask weight and image value in this position. Store the minimum in the current pixel location. After that, repeat this procedure with the background mask, and in backwards scanning order.

The common weights to use in the 2D case are shown in table 3.2. 'X' means that this position is not used. Weights for the 3D case are shown in table 3.3.

1	1	1
1	0	X

a. Forward mask

X	0	1
1	1	1

b. Backward mask

Table 3.2: Structural elements for 2D distance transform

The integer weights a...e should be optimized so that they minimize the error we get from using integer-only distances. Optimized values for isotropic 3D pixel grids were calculated in Borgefors and Svensson (2001), and found to be a = 3, b = 4, c = 5, d = 3 and e = 7 for the masks in table 3.3. Using these weights results in the least difference between the exact transform and the two-pass transform described above.

e	d	e
d	c	d
e	d	e

a. Forward mask, $z = i$

b	a	b
a	∞	∞
∞	∞	∞

b. Forward mask, $z = i + 1$

∞	∞	∞
∞	∞	a
b	a	b

c. Backward mask, $z = i$

e	d	e
d	c	d
e	d	e

d. Backward mask, $z = i + 1$

Table 3.3: Structural elements for 3D distance transform

3.3.4 Morphological closing and opening

The opening transform is an erosion on the image followed by a dilation. Mathematically, we write $A \bullet B = (A \oplus B) \ominus B$. The opening has the effect of removing thin lines and other small details.

The closing transform is a dilation followed by an erosion on the image. Mathematically, we write $A \circ B = (A \ominus B) \oplus B$. The opening has the effect of filling small holes and reducing the roughness of borders.

3.3.5 Geodesic distance transform

With normal distance measures, the length of the shortest path between points p and q is the distance between p and q. This is also the case for geodesic distance, but with geodesic distance, there are restrictions on which paths are valid. As an example, consider the subway. The shortest distance a subway wagon must travel to get from point A to B, is not generally along the shortest line between these points, the wagon must follow the tracks.

Mathematically, we can express the geodesic distance $d_A(p, q)$ between two pixels p and q in the connected set A as the minimum of the length $\mathcal{L}$ of the path(s) $\mathcal{P} = (p_1, p_2, \ldots, p_l)$ that join p and q and are included in A Sollie (2003) as (3.6).

$$d_A(p, q) = \min \{\mathcal{L}(\mathcal{P} | p_1 = p, pl = q, \text{and} \mathcal{P} \subseteq A\} \tag{3.6}$$

We will later want to calculate the geodesic distance from a part of an image, called a marker set, with path-restrictions, called a geodesic mask. The geodesic distance function calculated from a marker set Y included in a geodesic mask A, can be calculated by successive dilations of Y restricted

in growth by A Sollie (2003). Everything outside A gets infinite distance, to reflect that paths going outside A are not to be considered.

3.4 Segmentation

Segmentation can be defined as the partition of a digital image into multiple regions (sets of pixels), according to some criterion. This section explains various topics of segmentation that we will make use of in later chapters.

3.4.1 Thresholding

Thresholding is a method of segmentation. Basic thresholding is done by visiting each pixel (or voxel) in an image, and set the pixel to v_1 if its value is above or equal to a given threshold value and to v_2 if its value is below the threshold, where v_1 and v_2 are constants denoting certain intensity values. The values $v_1 = 255$ and $v_2 = 0$ are often chosen for 8-bit gray scale images.

There are essentially two main types of thresholding. In *global thresholding* the image is thresholded with a constant threshold value. In non-global or *adaptive thresholding*, the threshold value of a pixel depends on the intensity values in the area around that pixel, optionally including that pixel's value. Both global and adaptive thresholding-algorithms can in principle be fully automated.

One common way of finding the threshold automatically is Otsu's method (Otsu 1979). A recent, faster method with identical results exists (Lin 2003), but Otsu's method is fast enough in most situations on modern workstations.

Another classic and quick way of finding the optimal threshold value is to use the isodata method, often called "The Calvard-Ridler approach" (Ridler and Calvard 1978).

Both methods assume that the histograms have two peaks with each peak corresponding to a normal distribution.

3.4.2 Calvard-Ridler method

The Calvard-Ridler algorithm is given below (Sonka et al. 1999):

1. Assuming no knowledge about the exact location of objects in the image f, consider as a first approximation that the four corners of the image contain background pixels only and the remainder contains object pixels.

2. At step t, compute μ_B^t and μ_O^t as the mean background and object gray-level, respectively, where segmentation into background and foreground at step t is defined by the threshold value T^t determined in the previous step.

$$\mu_B^t = \frac{\sum_{(i,j)\in\text{background}} f(i,j)}{\#\text{background pixels}}$$

$$\mu_O^t = \frac{\sum_{(i,j)\in\text{foreground}} f(i,j)}{\#\text{foreground pixels}}$$

3. Set

$$T^{(t+1)} = \frac{\mu_B^t + \mu_O^t}{2}$$

   $T^{(t+1)}$ now provides an updated background-object distinction.

4. If $T^{(t+1)} = T^t$, halt, otherwise return to step 2.

The first step in the algorithm is sometimes changed to use a random starting value for the threshold. As an improvement to the algorithm above, the image histogram can be searched instead of the entire image, the T^ts can be stored as real numbers and the comparison in the last point be changed to ' If $|T^{(t+1)} - T^t| < \epsilon$, halt, otherwise return to step 2. ', where ϵ is the tolerance, a low value like e.g. 0.01. If this latter improvement is used, care must be taken to avoid an infinite loop when the difference between the Ts is not converging, but oscillating in a range larger in size than ϵ.

3.4.3 Otsu's method

Otsu's method is a way of finding the threshold automatically in an image where there are two potentially overlapping normal distributions (Otsu 1979). In statistical terms, it is based on minimizing the within-class variance and maximizing the between-class variance with respect to gray-level. It can be shown that this is equivalent to solving equation (3.7), where $\omega(k) = \sum_{i=1}^{k} p_i$, p_i is the normalized frequency of level i in the histogram, i.e. the probability of i, and $\mu(k) = \sum_{i=1}^{k} ip_i$, and $\mu_t = \mu(L)$, where k ranges from 1 to L.

$$T_{\text{opt}} = \max_k \frac{[\mu_t\omega(k) - \mu(k)]^2}{\omega(k)[1-\omega(k)]} \tag{3.7}$$

3.4.4 The watershed transform

The watershed transform is a method for segmenting an image. The image is considered a topographical surface. High pixel values will be peaks or "mountains" in this view, and in between mountains there will be valleys.

An analogy to water rising (immersion simulations) from the deepest valleys to the highest peaks is used. When the water rises, it will fill bays, or "holes" in the topographic surface. Bays of water are created as the water rises. These bays are the end result of the transform and correspond to regions in the image.

At water-level k, a certain number of bays exists because of the flooding. All pixels in those bays have value k or less. The water rises to level $k+1$. All pixels with values $k+1$ either belong to an existing region or is a new region. Each region has its "geodesic influence zone", which is an area around that region. Pixels in this zone belong to that region. Pixels that lie in influence zones are added to the influencing region. Others create new regions.

The geodesic influence zone $IZ_A(K_i)$ of a region in A is the locus of points of A whose geodesic distance to that region is smaller than their geodesic distance to any other region (Sollie 2003).

The watershed transform was first presented in Digabel and Lantuejoul (1978) and was presented in a more modern version in Vincent and Soille (1991).

3.5 Susan smoothing

SUSAN (Smallest Univalue Segment Assimilating Nucleus) is a nonlinear smoothing filter. It reduces image noise while trying to preserve image structure, i.e. not smooth image edges, that is borders between regions.

The *SUSAN* routine was invented by Smith and Brady (1997), who found it to provide a good quality to speed ratio compared to competing routines. The SUSAN principle can be used as a basis for feature-detection and structure-preserving noise reduction. The structure is preserved by only smoothing regions with low variance in intensity.

The original SUSAN equation is (3.8). The 3D extension is trivial and given by equation (3.9) (Skocir et al. 2002), where J is the filtered image and I is the original image. In the 2D case $r = \sqrt{i^2 + j^2}$, while $r = \sqrt{i^2 + j^2 + k^2}$ in the 3D case. The area (volume) that SUSAN considers before making a decision on current pixel intensity is given by a spatial threshold σ and a brightness threshold value. The brightness threshold, T, should be greater

than the noise level and less than the contrast of edges to be preserved. The area taken into consideration when smoothing is at maximum the area given by the spatial threshold, and is reduced to the pixels within the mask that is within the brightness threshold, measured from the current pixel. This (potentially) reduced area is called the "Univalue Segment Assimilating Nucleus".

$$J(x,y) = \frac{\sum_{(i,j)\neq(0,0)} I(x+i,y+j)e^{\frac{-r^2}{2\sigma^2} - \frac{(I(x+i,y+j)-I(x,y))^2}{T^2}}}{\sum_{(i,j)\neq(0,0)} e^{\frac{-r^2}{2\sigma^2} - \frac{(I(x+i,y+j)-I(x,y))^2}{T^2}}} \tag{3.8}$$

$$J(x,y,z) = \frac{\sum_{(i,j,k)\neq(0,0,0)} I(x+i,y+j,z+k)e^{\frac{-r^2}{2\sigma^2} - \frac{(I(x+i,y+j,z+k)-I(x,y,z))^2}{T^2}}}{\sum_{(i,j,k)\neq(0,0,0)} e^{\frac{-r^2}{2\sigma^2} - \frac{(I(x+i,y+j,z+k)-I(x,y,z))^2}{T^2}}} \tag{3.9}$$

3.6 Measurements

This sections explains concepts related to the measurements we will make in Chapter 6.

3.6.1 Pore and fiber chords

Pore chord measurements are a way of assessing information about the geometry of the void phase by measuring the pore chords in the structure. A pore chord is the local distance between two solid elements in the material (figure 3.3).

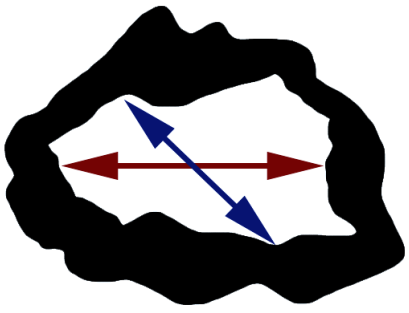


Figure 3.3: Pore chords

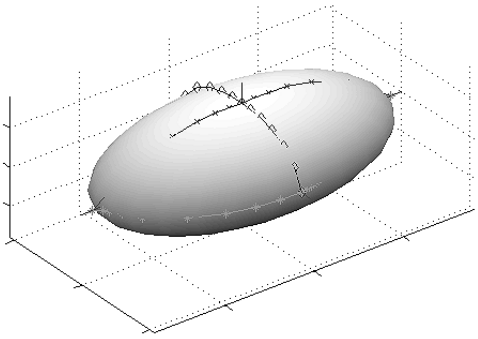


Figure 3.4: An equivalent pore ellipsoid

Pore chords can be measured in any spatial direction, and gives a measurement on the local one-dimensional extension of the pore volume. Mean pore height and width can be estimated using pore chords.

When average pore chord length (also called mean free path) is determined for all three principle axis, a visualization of this information as an ellipsoid, called the equivalent pore ellipsoid (figure 3.4), can be obtained. The equivalent pore ellipsoid has proved useful to describe differences between porous materials (Holmstad et al. 2003, Holmstad 2004). The oblongness of this ellipsoid, or its ellipticity is also known as the *structure anisotropy*, and it is measured separately in each principal direction.

A *fiber chord* is the local distance between two porous regions in the material, and the *mean fiber chord* is thus a compliment of the mean pore chord.

3.6.2 Principal component analysis (PCA)

Principal component analysis, also known as the *hotelling transform*(Gonzalez and Woods 2001, p.678) is a linear transformation that chooses a new coordinate system for a given dataset, such that the greatest variance by any projection of the data will lie on the first axis, the second greatest on the second axis etc. The first axis is called the *first principal component* of the data.

The principal components can be calculated by finding the *eigenvalues* and *eigenvectors* of the *covariance matrix*. The covariance matrix for three dimensions is shown in equation (3.10), where

$$\begin{aligned} S_{xx} &= \frac{1}{n}\left(\sum x^2 + \frac{(\sum x)^2}{n}\right) \text{and} \\ S_{yz} &= S_{zy} = \frac{1}{n}\left(\sum yz + \frac{\sum y \sum z}{n}\right), \end{aligned}$$

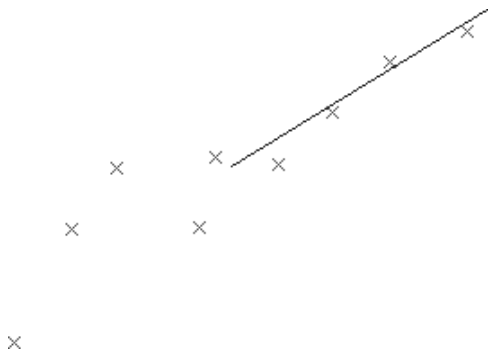


Figure 3.5: Example of principal component in a point scatter plot

where x,y and z denote the positions of the n voxels belonging to the object in the image. The remaining terms are evaluated using the same schemes.

The *eigenvalues* and *eigenvectors* can be extracted by a technique called *eigenvalue decomposition*. This involves finding the solutions to $A\mathbf{v} = \lambda\mathbf{v}$ where $\mathbf{v}$ is called an eigenvector of the matrix A and λ is its associated eigenvalue. The eigenvalue associated with the largest eigenvalue is the principal component of the data, and corresponds to the dimension that has the strongest correlation in the dataset.

Geometrically this can be interpreted as the direction that best describes the orientation of a set of disjoint point samples from its centroid and can easily be applied to voxel structures as well (figure 3.5).

$$\begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix} \tag{3.10}$$

Chapter 4

Initial segmentation

The separation of fiber and air in the volume is often called an initial segmentation in the literature (Holen and Hagen 2004, Holmstad 2004, Bache-Wiig and Henden 2004). We will for practical reasons extend this term to include the separation of the phase of interest from other phases, so that we can use the same terminology on both composites and paper volumes.

The initial segmentation is used for measuring structural properties of the material, which are of special interest (Huang et al. 2002). Additionally, it can be used for further segmentation of the volume, to achieve a segmentation into individual fibers (Holen and Hagen 2004, Lunden 2002, Yang 2001, Holmstad 2004). These two goals are not necessarily in conflict, but they are different in that the former requires a physically correct binarization, while the latter does not. However, because we want both a physically correct initial and individual segmentation to do measurements from, it can be practical to use the physically initial segmentation as a starting point.

This chapter explains our method for initial segmentation, and its relation to methods previously suggested in the literature.

4.1 Previously suggested methods

Methods for the binarization of absorption-mode paper tomograms have been suggested by Bache-Wiig and Henden (2004),Holen and Hagen (2004), Holmstad (2004) and, more recently, Rolland du Roscoat et al. (2005).

The binarization of phase-contrast tomograms have historically been difficult. In volumes from previous years (< 2003), borders between regions were difficult to notice, or simply not there at all. Methods for binarization of such phase-contrast paper tomograms have been suggested by Kvestad

(2002) and Antoine et al. (2001). We will focus on the absorption-mode methods in this paper.

Common for all the mentioned absorption-mode methods is that they start with a smoothing filter to reduce noise in the volume. A binarization is then done by thresholding and/or region growing, and often ends with the removal of small regions.

The smoothing is done by averaging each point in each tomogram with neighboring tomograms in Holmstad (2004), which blurs edges, by using Susan 2D smoothing in Holen and Hagen (2004), which ignores information from neighboring tomograms, and by using Susan 3D smoothing in Bache-Wiig and Henden (2004). In Rolland du Roscoat et al. (2005) and Yang (2001), anisotropic diffusion was used for smoothing. Tests done by Holen and Hagen (2003) on absorption-mode paper tomograms showed that anisotropic diffusion and Susan smoothing gave equally good results, and that the Susan method was the fastest. Tests done in Bache-Wiig and Henden (2004) showed that there is a significant difference between 2D and 3D Susan smoothing of a volume, and that the 3D version is preferred if further 3D processing is to be done on the volume.

The thresholding in Holmstad (2004) and Holen and Hagen (2004) is done by manually selecting a global threshold value, and in Bache-Wiig and Henden (2004) by finding the global threshold value automatically by the Calvard-Ridler method, while Rolland du Roscoat et al. (2005) uses a global threshold to find seeds for region growing. A different approach is chosen in Yang (2001), where the binarization is done by thresholding with indicator kriging (Oh and Lindquist 1999), which is a statistical, a priori based approach.

As the intensities in the paper tomograms vary not only based on the object being imaged, but also on the position in an unknown manner a different procedure than global threshold is required (Rolland du Roscoat et al. 2005). Region-growing using a stopping-rule that depends on the objects being grown was successfully used in Rolland du Roscoat et al. (2005).

For a more thorough summary, see Bache-Wiig and Henden (2004).

The literature on initial segmentation of fiber-reinforced composites seems to be limited to Jain and Dubuisson (1992), who examined defects and damages to composite samples. Their segmentation resulted in an image that showed these defects and damages. Their image processing problem and images were quite different from ours, and we cannot use their methods in our task because of this.

4.2 Suggested method

With this method we try to achieve both a good starting point for further segmentation and a close to physically correct binarization.

Preparations. The images are converted to 8-bit at ESRF, as explained in Section 2.1. They are stored in raw format, 256 slices in each file. A suitably large, centered sub-volume is extracted from each raw file, and these are concatenated to form one, large volume file, so that there will be no border-effects from merging the volume-files later. Slices containing no fibers are removed from the start of the volume. Slices with post-it parts are removed from the end of the volume. The result is a volume where no slices contain post-it parts and all slices contain fibers.

Smoothing. The initial segmentation starts with 3D noise reduction using Susan smoothing. The smoothing is done three times, iteratively. The brightness threshold is lowered after each run, as the intensity range of the noise is lessened. We found that a division factor of 2.0 in brightness threshold, e.g. $50, 25, 12$, between runs produced good results on the paper volumes. The division factor must depend on the contrast in the image, with low division factors being used on images with low contrast. The composites have lower contrast, and we chose a division factor of 1.5 for the composites because of this.

Example results of the smoothing of S8a[1] are shown in figure 4.1.

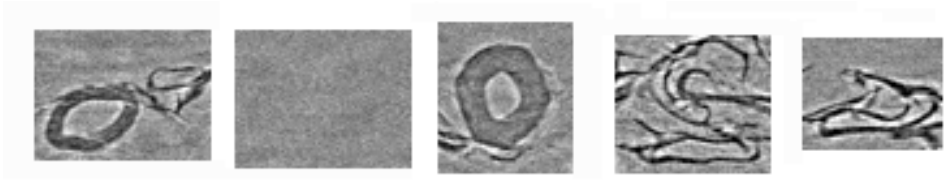
We found that the separability between the two distributions improves as a result of the smoothing of the volume (figure 4.2).

Adaptive thresholding. There are many different ways of doing adaptive thresholding. A simple method we found to preserve borders slightly better than global thresholding is described below (figure 4.3).

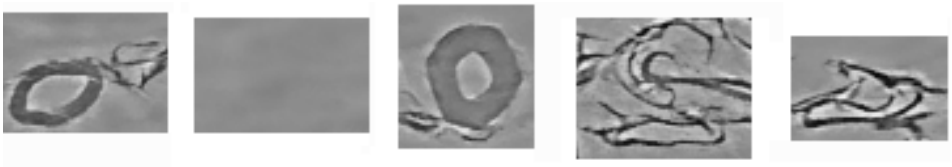
$$T_{x,y,z} = T_{\text{base}} - \omega \frac{\sum_{i,j,k \in N_{x,y,z}} I(i,j,k) - T_{\text{base}}}{n} \tag{4.1}$$

The threshold value is decided on a per-pixel basis. The threshold value of a pixel with coordinates x,y,z, is calculated from equation (4.1), where *base* is the base threshold, found manually, or found by a fitting automatic method,

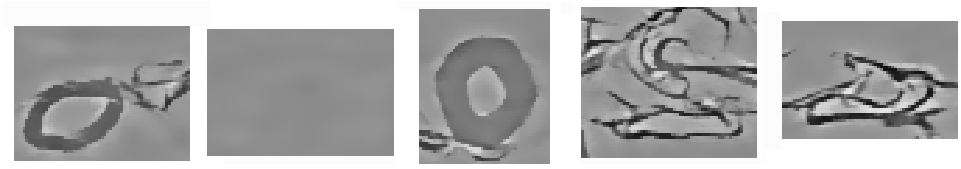
[1]The smoothing was done on a 16-bit version of S8a. Using a 16-bit volume shows the shape of the histograms better, as the number of different intensity levels is greater than in the 8-bit case.



(a) Examples, original



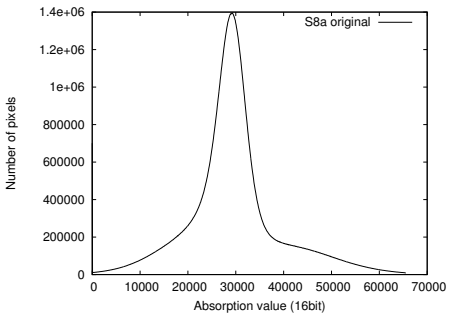
(b) Examples, smoothed once



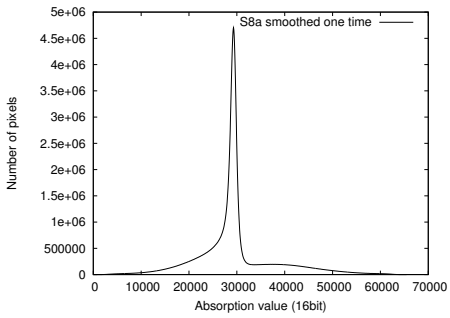
(c) Example, smoothed twice

(d) Examples, smoothed three times

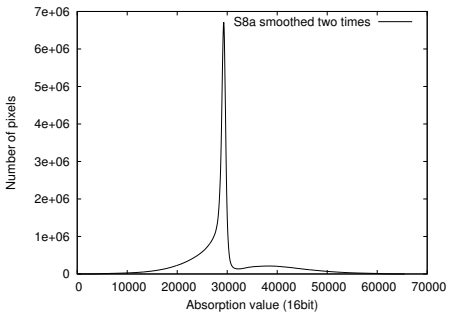
Figure 4.1: Results of Susan3D, brightness threshold = 12000, 6000, 3000 and σ = 2.0 (16-bit data)



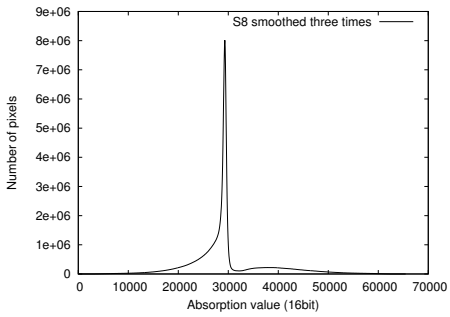
(a) Original



(b) Smoothed once



(c) Smoothed twice



(d) Smoothed three times

Figure 4.2: Histograms of Susan3D, brightness threshold = $\{12000, 6000, 3000\}$ and $\sigma = 2.0$

Figure 4.3: Global thresholding (top row) versus adaptive thresholding

ω is an ad-hoc weighting parameter, I is the image, $N_{x,y,z}$ is the neighborhood of pixel (x, y, z) and n is the number of pixels in the neighborhood.

The net effect of using this equation is that pixels in dark areas will get a higher threshold, while pixels in bright areas will get a lower threshold. In images where intensity decreases in the direction from center of object to object border will this way of thresholding result in a slightly better segmentation of the object borders. It will also help to reduce small bright spots in dark areas. Whether these properties are of value depends on the actual image processing problem.

We tried different methods for finding T_{base} automatically. Neither Otsu's method or the Calvard-Ridler (isodata) method gave bad results, but we found the intensity level corresponding to the deepest point in the valley between the two peaks in the histogram to be a better base threshold in our cases. We found this level from searching the global histogram manually. The two methods differed in that the isodata method often gave a lower threshold value than Otsu's, and sometimes lower than our manual found threshold, while Otsu's always gave a higher threshold value than what we found manually. We found that we could reduce the threshold value slightly from the base, i.e. move it closer to the background mean, when using this procedure, as the number of small regions added is less than in the case of global threshold at a reduce threshold value.

The threshold can alternatively be found by the method described in Section B.1 on page 113, which uses a porosity value from physical experiments on the material to decide on threshold value.

A small neighborhood ($3 \times 3 \times 3$) and a low local weight (10%) was used.

Figure 4.4: Examples from the binarized S8a

Isolated, small cluster removal. Three-dimensionally isolated pixels in the foreground and the background are removed. This is done to speed up the next step, which is to remove regions smaller than a certain volume.

Regions with a volume of less than $216\mu m^3$ are removed. This size is, according to Holmstad (2004), a good trade-off between preservation of structural features and removal of noisy elements in x-ray tomograms of paper.

The removal of non-isolated pixels is done as suggested by Yang (2001), and implemented by Holen and Hagen (2004). The volume is scanned from top to bottom, and for each foreground pixel, a flood fill is performed. When the flood fill finishes, the size of the volume filled is checked. If the size is below a certain threshold, the filled pixels are colored with the background color. The memory requirements are reduced by marking visited pixels to avoid adding them to the queue more than once. A breadth-first search strategy is employed when flood filling.

A sample result of the removal algorithm is shown in figure 4.5. Note that because the input is a 3D volume, small regions in a 2D slice may remain.

This completes the segmentation for normal volumes. For volumes with much ring-noise, the process continues to the next step.

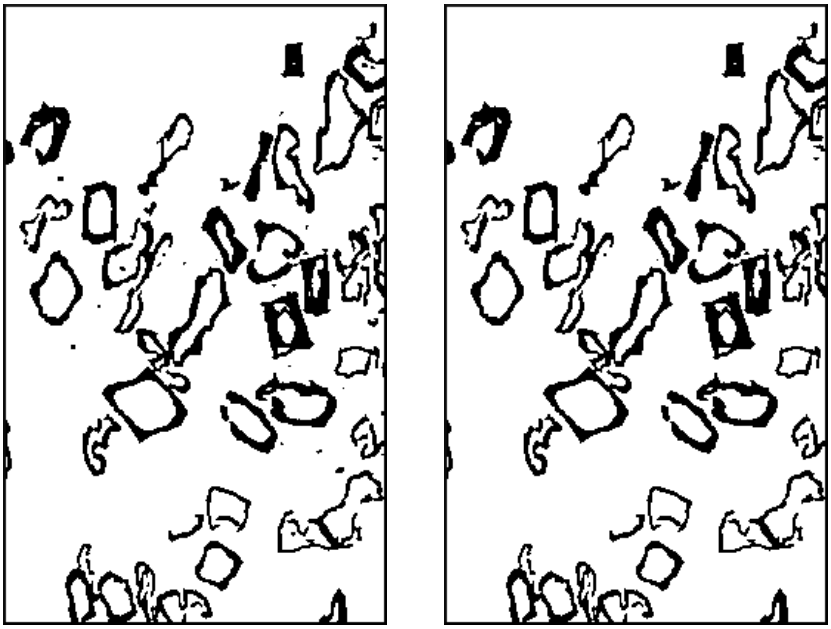
Dilation, masking and new iteration. To arrive at a more physically correct binarization, we want to include a bit more of the fibers than we have at the present point in the method. As a way of including a bit more of the borders and interiors, the foreground is dilated a small number of times and used as a mask on the smoothed image.

A new 3D adaptive threshold is then performed, this time with a lower threshold than earlier. This has the effect of re-introducing small regions that were removed, but should not have been, and reduces the break-up of borders.

Isolated, small cluster removal is then repeated.

This completes the initial segmentation.

More images of binarized volumes can be found in Chapter 6. A sample



(a) Without removal (b) With removal

Figure 4.5: Illustration of the removal of small regions

highlighting the differences between this approach and that of Holen and Hagen (2004) can be seen in figure 4.6. This image was created by first subtracting the HH–binarized volume from the same volume binarized by the new method, and making all points in this image black. This difference images was added to a grey version of the HH–binarized volume, meanining that the black dots and areas in the resulting image were missing from the HH–binarized volume, but present in the new volume.

Figure 4.6: A comparision of two binarization methods

4.2.1 Algorithm

The algorithm for binarization then becomes:

1. Reduce noise while preserving structure (Susan3D). If necessary, multiple runs of Susan3D is performed

2. Find the optimal threshold by finding the lowest point in the valley between the two peaks in the histogram, or by the porosity method
3. Do 3D adaptive thresholding with the optimal threshold as base
4. Remove isolated pixels in 3D
5. Remove regions smaller than a certain limit
6. For noisy images: Mask the smoothed image from step 1 with the dilated image from step 6. Repeat steps 2 to 5 once more, this time use a lower threshold

The necessary programs have been implemented in Java/ImageJ. Details can be found in Section 7.5 on page 90.

4.3 Discussion

While there is reason to believe that the new algorithm is better than that of Holen and Hagen (2004), as the smoothing uses 3D data and the thresholding is adaptive, it is difficult to say how it compares to the procedure of Rolland du Roscoat et al. (2005). It is the authors' opinion that region-growing is a good solution to this binarization problem, and Rolland et al. showed good results in their article. While our procedure also gives good results, the Rolland-procedure has the benefit of being less complex, and requires fewer input parameters. These two procedures should be compared to each other. We have not had the time to do so ourselves, but we will release all our software to the public domain, enabling the Rolland-team or others to do the comparison, if they so wish.

Chapter 5

Segmentation of paper fibers

Many useful measurements require individual fibers to be identified and separated from the bulk material. We have developed a set of tools to aid the identification of such fibers. This chapter will present and explain these, as well as the prior work they are based on.

5.1 Previously suggested methods

To our knowledge, the only previous efforts to identify and trace paper fibers in a network have been described by Aronsson (2002), Holen and Hagen (2004) and Lunden (2002). In addition Yang (2001) segmented other types of fibers.

The extraction of blood vessels from medical images shares many characteristics with fiber network segmentation. One of these methods was adapted to fibers in Aronsson (2002, p.42), based on ordered region growing (see Section 5.1.1).

5.1.1 Aronsson (2002)

Aronsson has suggested and implemented both 2D- and 3D-based methods for segmenting natural paper-fibers from three-dimensional images in Aronsson (2002) These volumes were obtained by grinding through the material with a microtome, and photographing each slice using a Scanning Electron Microscope (SEM). The quality of the slices was generally high, but the method resulted in non-isotropic voxel dimensions, complicating the segmentation process.

The first attempts by Aronsson were heavily based on manual input, where each image slice was treated separately, having the user select a different segmentation method on each slice. Lumens were filled based on how collapsed or cracked they were in the image. Notably 2D snakes based on a Gradient Vector Field were applied on lumens where larger holes existed. In this process the user would have to manually select a method and mark a seed in each of the visible fiber walls.

Aronsson also proposed a fully automatic method of identifying circular fiber cross-sections in paper (Aronsson 2002, p.117). The method can be summarized as follows:

1. Mark lumen candidates by identifying maxima in the distance transform
2. Filter these candidates based on roundness and area
3. Obtain the local distance transform inside a region bounded by a rectangle slightly larger than the lumen candidate, and estimate average wall thickness based on the average distance transform values
4. For all image pixels, label each pixel as belonging to the detected lumen candidate closest to that pixel by applying a modified forward distance transform
5. Remove labels from all pixels that have a thickness of more than 150% of the average wall thickness

Aronsson (2002, p.41) concludes that this method had mixed results throughout the volume, and was not suitable for a full 3D segmentation. The main problem stated was the extremely tedious manual editing imposed by having to re-seed each lumen on each slice in the volume.

Aronsson also suggested two different approaches to fully 3D based segmentation:

3D local vector path estimation. This approach is based on seeding some fibers, and then automatically tracing these fibers throughout the volume by estimating a local direction based on the distance transform (Aronsson 2002, p.41). A major problem with this approach was that the path often diverged through cracks in the fiber because the direction estimation was not stable enough. According to Aronsson this method can be useful if an improvement to the local fiber path estimation is made.

We have attempted an improvement to this method, but ended up not using it, as we tried a different approach than tracking by vector path. In hope that it can be useful to others, we have included it in the appendix, Section B.2.

Ordered region growing. The main difference between this method and the vector path estimation vector is that the ORG can recover if it gets lost. The problem with the vector path estimation method is solved by requiring a user to select both a starting and ending-point of a fiber's lumen.

Starting at one end-point, boundary voxels are added if they are accepted by an evaluation function f (5.1), where DT($\mathbf{v}$) is the value of the current voxel $\mathbf{v}$ in the 3D distance transform of the volume, $\mathbf{v}_{\text{start}}$ is the start seed and $\mathbf{v}_{\text{end}}$ is the end seed.

$$f(\mathbf{v}) = \text{DT}(\mathbf{v}) + \frac{1}{\|\mathbf{v}_{\text{end}} - \mathbf{v}_{\text{start}}\|}(\mathbf{v} - \mathbf{v}_{\text{start}})(\mathbf{v}_{\text{end}} - \mathbf{v}_{\text{start}}) \tag{5.1}$$

This is repeated, until the other end-point is found in the search tree. Each node has only one parent node, so it is possible to backtrack from the destination point a unique path to the start-node, giving us a complete skeleton of the lumen.

The lumen was reconstructed from the ORG path, by using the 3D SeparaSeed method (Tizon and Smedby 2002), somewhat similar to the well known watershed transformation (see Section 3.4.4), but designed to be much faster by using a local mask and not requiring voxel sorting (Aronsson 2002).

Wall segmentation was done by generating a distance transform from the lumen voxel region. Then a histogram is created for distance values within the foreground region of the original data-set. Plotting this histogram shows how many voxels in the fiber wall that exists at a given distance from the lumen. Given that the fiber wall has an approximately constant thickness, a sharp drop in this histogram gives an estimate for the average wall thickness. Then a distance transform from the lumen is made, and all foreground-pixels reasonably within the average wall thickness are labeled.

None of the software created in Aronsson (2002) is freely available to other scientists.

5.1.2 Lundèn (2002)

Lundèn's paper is based on Aronssons work, although the two papers were released at about the same time. It represents an implementation of methods from Aronsson (2002) in Matlab, with some extensions. The original implementation was in C, C++ and some Matlab.

Lundèn identified fibers by their lumen, as Aronsson did. In addition a method for finding collapsed lumens was suggested. First non-collapsed

fibers were found and removed. Then seeds for collapsed fibers were found by thresholding. A region growing procedure was performed from these seeds with the edges in the original image, found by a Canny edge detector, as a stopping criteria. The software was semi-automatic in that candidate fibers were found automatically and the user selected which of these were true fibers for each slice in the volume.

The program presented the user with approximately twice as many false candidates as true candidates for each image slice, and used 1–4 minutes per slice to find the candidates. The selection of true candidates took about 25 seconds. The success-rate for non-collapsed fibers was above 90%, and about 50% for broken and collapsed fibers (Lunden 2002, p. 63). These percentages are dependent on the images being segmented.

The software is the property of STFI and not freely available to other scientists.

5.1.3 Holen and Hagen (2004)

Individual fiber segmentation was achieved by Holen and Hagen (2004) by tracking lumens. Lumen coordinates for each fiber was given by user input. Bulk segmentation is used as the basis for the tracking. Domain rules were used in the tracking procedure.

The method was based on the assumption that fibers were aligned along a nearly straight line (Holen and Hagen 2004, p.7). Only fibers with a lumen are trackable through this method and fibers that loose their lumen after time (e.g. walls cave in) are not completely tracked.

A significant amount of user-interaction is necessary, because each fiber has to be seeded at least once (twice or more if the fiber loses its lumen for a couple of slices), however the method is much more practical than the 2D-based method suggested by Aronsson since the procedure is fully automated after the lumens have been seeded. In contrast, Aronsson required the user to reseed each lumen on each slice, as well as choosing a method for crack healing manually.

A large part of the tracking procedure concerns crack detection and making sure the cracks do not produce an erroneous segmentation. Wall labeling was based on watershed, with distance as segmentation criterion. In addition, a thickness estimate was calculated on the first slice, used to limit the wall growth within reasonable limits.

The method was tested on two paper samples, TMP-based and laboratory-based newsprint. A success-rate of 164 of 178 manually seeded fibers was reported for the former and 150 of 166 fibers for the latter.

The domain rules and procedures used are based on those in Aronsson (2002).

5.1.4 Yang and Lindquist (2001)

Yang proposed a set of techniques for doing structural analysis of a three-dimensional image of a fiber network in Yang (2001) and Yang and Lindquist (2000).

The procedure assumes that fibers have a solid core, i.e. they have no lumen. The result of the medial axis transform on hollow fibers is thin shells (surface remnants). The surface remnants are dealt with when post-processing the medial axis network. They are replaced with the shortest path through the surface remnant to joining fiber-paths that includes the voxel of the surface remnant that is closest to the center-of-mass of that surface remnant (Yang 2001, p. 48).

The first stage is initial segmentation by thresholding. The second stage is segmentation into individual fibers, based on the medial axis transform and tracing of fibers. The third and last stage is post-processing of this medial axis network to arrive at a representation of the individual fibers by their medial axes.

When segmentation is complete, measurements are done on the resulting fiber network.

5.1.5 Vessel-extraction from medical images (2004)

There has been a large amount of research on the extraction of blood-vessel networks from images. This task is similar to the one of individual segmentation of hollow fibers.

The major differences lay in that fibers can collapse, lose their lumen, and that they can bend at sharper angles than blood-vessels. Additionally, fibers do not branch, while blood-vessels do. A fiber's walls can be pushed together, separating the lumen into n disjoint regions. These regions often merge later on. This is somewhat similar to branching.

A review of vessel-extraction techniques is presented in Kirbas and Quek (2004). We recognized ideas and methods in this paper from our own work and the literature on fiber segmentation, but found none of them to be easily applicable to our problem.

5.2 Suggested methods and tools

Based on previous efforts and our own suggestions, we have implemented a set of tools to successfully label single fibers in the data-structure. Most of the prior methods were based on tracing lumens from user-selected seeds, and we have used this approach as well. We have divided the fiber identification into a two step process and made the lumen- and wall-labeling methods interchangeable components. This has enabled us to experiment with a number of different approaches to fiber segmentation within the same framework (see Chapter 7).

While the ORG–approach (see Section 5.1.1) seems very promising, the results presented indicated that the approach by Holen and Hagen (2004), from now on called the HH–approach, was more robust. In Holen and Hagen (2004, p.43) 60-75 % of the fibers could be identified in a sample, while Aronsson picks up only about 10-20%. However it must be noted that these percentages is neither based on the same material or data quality, and hence cannot be used for direct comparison.

The ORG–approach requires that both ends of a fiber is manually identified, something that is impossible to perform automatically, since it would require that the lumen had already been identified. The HH–approach requires only a single point to be identified for each lumen, making it possible to at least partially automate the process.

Another argument in favor of the HH–approach, is that it handles collapsed lumens in a fairly straightforward manner, where Aronsson to our knowledge does not attempt to address this problem. The final argument in favor of the HH–approach is that the source code and documentation have been fully available to us.

Based on these findings we have decided to use the HH–approach as a basis for lumen tracking. We believe that using this approach to track fibers along all three principal directions in combination with a true 3D based wall labeling method (see Section 5.2.2), has enabled us to give fairly unbiased data to be used as the basis for measuring the fiber orientation distribution.(see Chapter 6).

As well as implementing the method in a computationally efficient manner, largely based on removing redundancy and region-bounding local mask-operations, we have created a practical tool ready for scientific use at PFI and other research facilities.

5.2.1 Lumen tracking

Our approach is outlined by the following procedure:

1. An intact lumen-slice is filled at any randomly selected position inside a fiber.
2. The corresponding lumen-pixels are then copied onto the background-pixels in neighboring slices, effectively masking out parts of the lumen now covered by walls (foreground pixels). If the previous lumen is entirely covered by foreground pixels, the process is stopped.
3. Cracks are detected and sealed if necessary using the method described in Section 5.2.1. If the crack-to-wall percentage gets above a certain threshold, stop.
4. The new lumen can now be constructed by filling up the remaining void.
5. Repeat from 2

This is a simple but efficient approach if the data is clearly defined and fibers are aligned in roughly the same direction. This assumption is unfortunate, but repeating the operation from the three primary orientations of the volume will enable us to pick up all but the most bent and collapsed lumens. Note that the

The main issues associated with this approach are how to handle cracks (step 2), and lumen collapse (step 5). Both issues and solutions are explained in the following paragraphs. Since the method requires manual seeding from several parts along the volume, we have also suggested and implemented a method for automated lumen detection in the final section in this chapter.

Cracks

Cracks in lumen-walls occur at random places, and makes leakage from a lumen to the surroundings a major problem.

The proposed solution in Holen and Hagen (2004) to detect cracks in a particular slice, was based on calculating the geodesic distance from the lumen identified in the previous slice, and then identify cracks by identifying a low number of occurrences of any particular distance label.

Aronsson suggested a different approach to the same problem, using 2D snakes, based on the Gradient Vector Field extension, to lock onto the fiber walls (Aronsson 2002, p.39). He reported that the method had stability issues, and this discouraged us from attempting this approach.

Figure 5.1: Example of detected crack from old lumen

Since few cracks tend to last for more than a couple of slices, we attempted several simpler approaches based on the idea that we would restrict growing the previously identified lumen across all consecutive cracked slices. However, we concluded that the first solution was the most accurate and stable one, and have adopted this solution for our final implementation.

Collapsed fibers

In many cases the traced lumen is split up into several parts. Usually this is the result of noisy data or faulty crack-detection, and we ignore all but the largest lumen part. However, there are some cases where a single fiber collapses and two separate regions should be kept. The collapses that can lead to a split lumen are illustrated by branch a and b in figure 2.9 on page 16.

Holen and Hagen (2004, p.28) noted that fibers tend to collapse around the middle of the original cross-section, resulting in two separate regions of roughly equivalent size. Using this as a criterion, we only keep track of two separate lumens if the two largest parts are roughly the same size.

5.2.2 Wall labeling

Having completely identified a lumen makes it possible to identify the surrounding lumen walls. This is complicated by the fact that many fibers share the same border and walls must be divided accurately between these.

The HH–approach was based on the assumption that fibers were aligned along a nearly straight line. We have included their suggestions as an option for wall labeling in our framework, and it can be useful when fibers are highly aligned in one direction. The main advantage with this method is that it

can bridge gaps between separated lumens, and hence make the wall labeling more accurate in such cases.

The method suggested by Aronsson have some clear advantages, since it is fully 3D based, and completely unbiased regarding fiber orientation given that all lumens have been accurately identified. We did not have access to all the details surrounding it, but have adapted a similar approach using geodesic distance transforms and generating histogram information on the amount of voxels added to each lumen at each distance. The main difference is that we estimate the wall size based on a lower threshold for the amount of added voxels, rather than attempting to detect a sharp drop in the histogram. This seems to make the method a little more robust in cases where wall sizes varies, and no sharp drop in the histogram can be found.

In figure 5.2(a) we have illustrated the problem using the HH–approach on our new data sets, showing that it does not handle fibers perpendicular to the tracking direction well. Figure 5.2(b) shows the result of our proposed wall labeling method, showing clear improvements to the segmented results. Note that white regions contain unlabeled material.

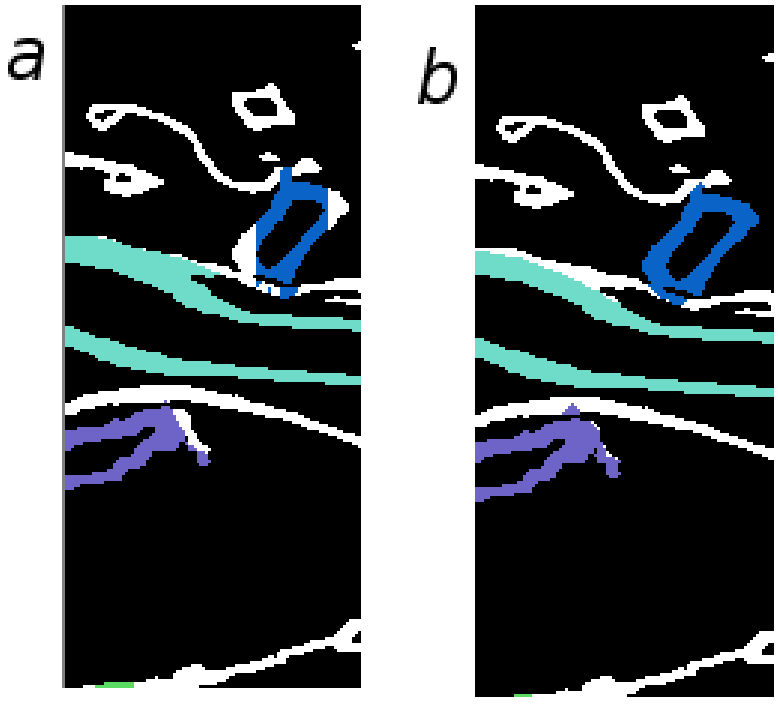


Figure 5.2: Figure (a) shows the view plane perpendicular to the tracking direction using a wall labeling similar to the HH–approach. Figure (b) shows the same example tracked using our proposed wall-tracking method

5.2.3 Method for wall-labeling

We start by searching the neighbors of any foreground voxels (corresponding to white pixels in figure 5.3) to see if there are any neighboring lumen-labels. If this is the case, we copy this label to the inspected voxel. This will effectively grow each lumen by one voxel into the foreground (wall)-region, in a similar fashion to a geodesic dilation (see figure 5.3b)

Now repeat the procedure n times with the following restriction: If the amount of voxels added to a lumen is less than a quarter of that added on the first iteration, mark this lumen as "complete" to avoid further propagation from this lumen. This restriction avoids uncontrolled growth into erroneous branches.

The last step is to remove lumens and unlabeled regions from the final image, as shown in figure 5.3e.

5.2.4 A method for automatic seed detection

The method presented in Holen and Hagen (2004) requires the user to manually select starting points for lumens, and explicitly mark collapsed parts as belonging to the same fiber. A similar approach is described by Aronsson (2002, p.38) in which the user selects marks at well-defined lumens in cross-sectional images to be used for lumen tracking. This applies to all three approaches suggested, including 2D-based segmentation, ordered region growing and 3D vector path estimation. Thus providing a way to automate, or simplify the identification of such lumens would possibly benefit all three segmentation approaches since they all require start-points to initiate the lumen search.

In Aronsson (2002, p.117) a fully automatic method for detection and labeling of clustered ring shaped objects was suggested. It is based on the idea that when searching for fibers it is easier to detect the lumen voids as circular regions rather than detecting the rings themselves. The method assumes that fibers have a constant wall thickness around the entire lumen, and that the lumen roughly has the shape of an ellipse.

The basic principle was based on detecting connected regions in the 2D slice, and labeling those that conformed to the following three criteria:

1. 50 <= lumen area <= 2000
2. 1 <= circularity <= 5
3. 0 <= average curvature < 15

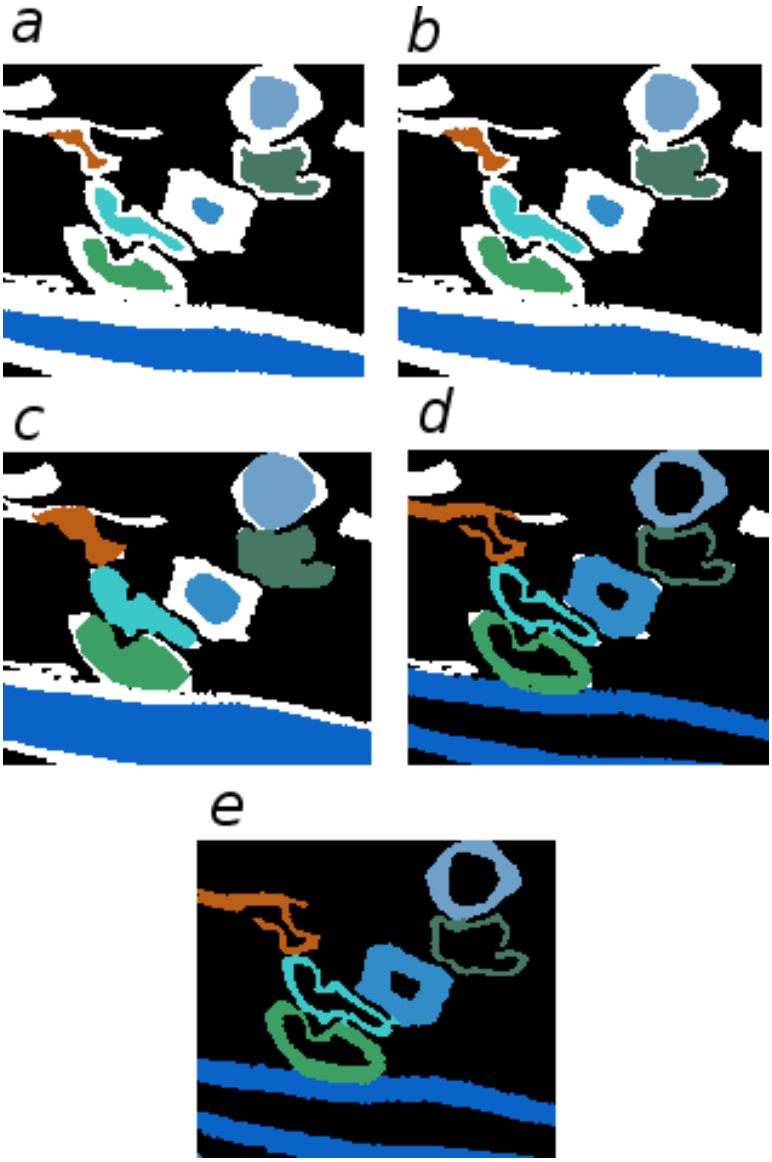


Figure 5.3: Example application of the progress during wall growth. (a) Shows the original lumen image (b)Shows the first iteration (c) The fifth iteration (d) The final result with original lumens removed (e) Final result with unlabeled voxels removed

The constants are clearly data and domain dependent, while the principle is applicable to our data as well.

Alternative criteria for lumen detection

We have implemented and tested the criteria above as well as bending energy, eccentricity, convexity and solidity (see Chapter 3 for definitions). We wanted to test bending energy because it quantifies the energy stored in the shape of the contour, and in biology, energy usage is often minimized. Convexity and solidity are both interesting descriptors because the lumens are mostly convex. The eccentricity is relevant for excluding very long, thin shapes.

The tests were done by selecting 70 lumens and 70 non-lumens close to evenly distributed from the volumes S8a, STFI7 and STFI9. See Section 6.3 for a description of these volumes. These volumes were chosen because we wanted to make measures on them later, and therefore segment them well.

We found that curvature and bending energy values did not distinguish easily between lumens and non-lumens. Because of this, we tried two different approaches for measuring them, both the new method of Mackworth and Mokhtarian (1988) and the classic method of Young et al. (1974). They did not differ much in terms of distinguishing between our two classes of shapes. There is a slight tendency for lumens to have higher curvature values than non-lumens, but it is difficult to select a separation line between the two classes. Smoothing the lumen borders using curve evolution with a σ of 3 did not improve the separability significantly. A rule saying that lumens have a curvature value above 0.8 will exclude some non-lumens, but also some lumens (figure 5.4). The bending energy is more sensitive to changes along the border than the curvature as it is the average, squared curvature, and tends to give higher values for non-lumens than lumens, but separates even less clearly between the two classes.

Eccentricity gave some more information. There is a clear tendency for lumens to have lower eccentricity values than non-lumens, but many non-lumens have small eccentricity values too (figure 5.6(a)). Some non-lumens have especially large eccentricity. In order to focus at the range of overlap, these are not shown in the plot. A rule saying that lumens have an eccentricity below 50 will exclude some of the non-lumens.

Convexity and solidity values, which is based on convexity, separated more easily between the two classes. A rule saying that convexity must be between 0.8 and 1.0 will exclude almost all non-lumens (figure 5.7(a)). Most of the lumens have a solidity value above 0.9 and below 1.1, but not all of them. There are some below, even as low as 0.65. A rule saying that solidity should

be above 0.65 will exclude some non-lumens, or alternatively, the threshold 0.9 will exclude most non-lumens and some lumens (figure 5.7(b)).

Size can be used as a classifier in combination with other descriptors in this case. Some non-lumens have especially large sizes. In order to focus at the range of overlap, these are not shown in the plot. We found almost all lumens to be between 40 and 4000 pixels in area, that is approximately 82–8200 μm^2.

Circularity can also be used to separate the two classes (figure 5.8). Some non-lumens have especially large circularity values. In order to focus at the range of overlap, these are not shown in the plot. A threshold at circularity value 4.0 will exclude a large amount of non-lumens. A lower threshold can be used to exclude more non-lumens, but that will exclude some lumens as well. A lower threshold at 0.5 can be used as well, as the lowest measured value for lumens were 0.7.

Additionally, we reject lumens with two or more internal regions, i.e. lumens that have bits of fibers inside them, as this is an indication that the lumen's shape makes it unfit for tracking.

Lumens that touch image borders are also rejected. This is because regions that are partly closed by a line or a corner instead of a ring shaped fiber wall are shapes that we cannot classify correctly using the descriptors above.

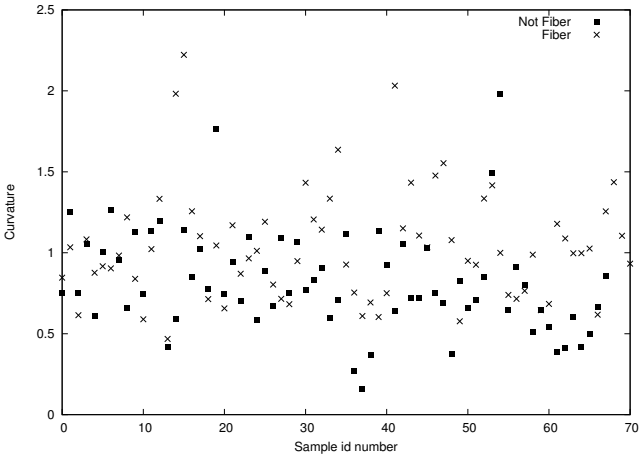
A summary of these observations is shown in table 5.1a. If we want to reduce the number of false positives, we can formulate the rules more strictly (table 5.1b.). This will have the effect of reducing the number of true positives as well.

Algorithm for lumen detection

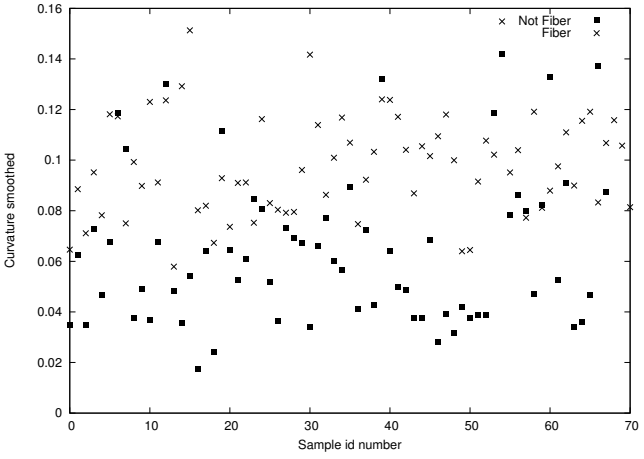
Based on the method in the previous section, we have automated some of the manual steps of the HH–approach (Holen and Hagen 2004).

The success of this method depends on the data it is applied to. Thresholds can be fine tuned to the data of interest. The result should therefore not be viewed as a final result, but as a good basis from which lumens can be added to and deleted from in order to achieve a complete lumen selection. If we define the success rate of the automatic detection as how large a percentage of lumens that can be used as input to our lumen tracking algorithm, we have a success rate from about 50–70%, based on our experiments. This means that the automatic detection is a good time saver, as it more than halves the manual work needed to select lumens.

The descriptors, and threshold rules above are easy to change in our fiber

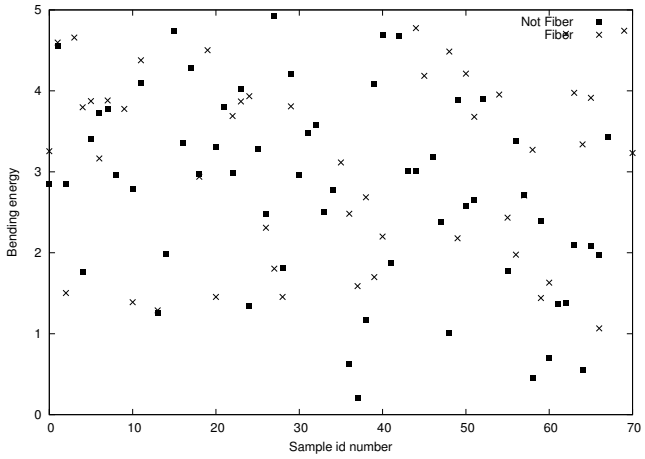


(a) Curvature

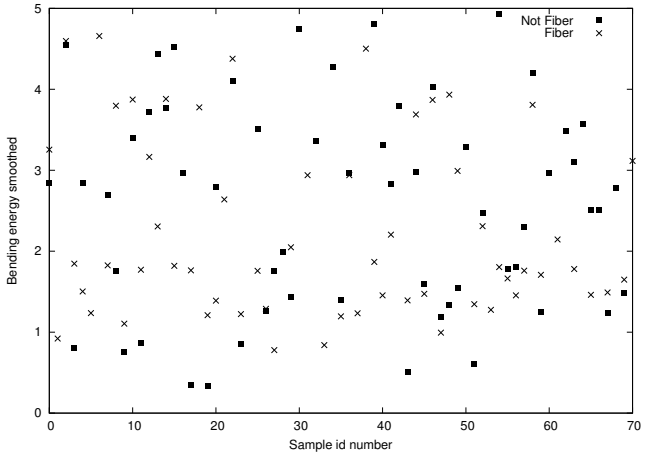


(b) Smoothed curvature

Figure 5.4: Plot of curvature values of regions

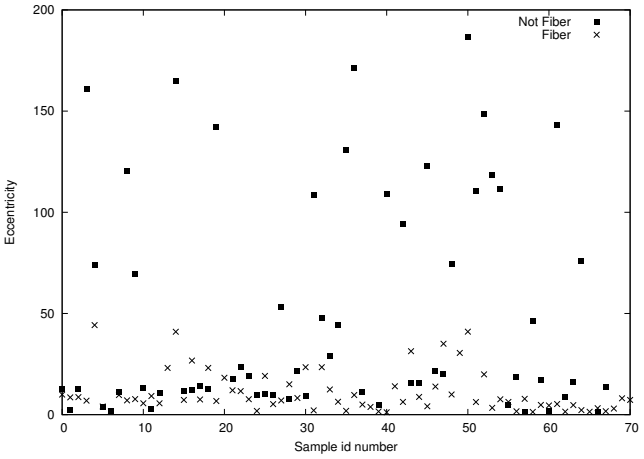


(a) Bending energy

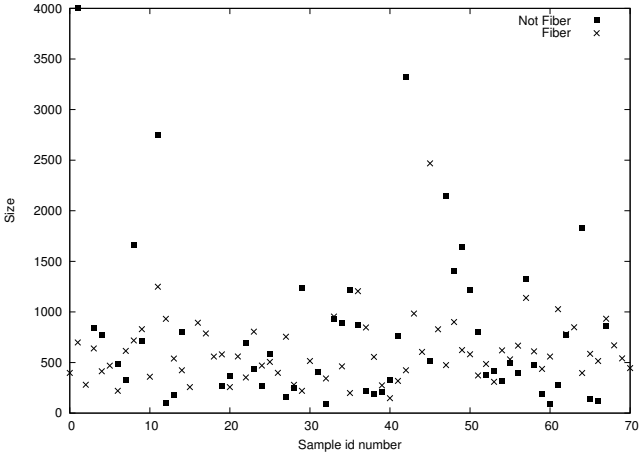


(b) Bending energy smoothed

Figure 5.5: Plots of bending energy values of regions

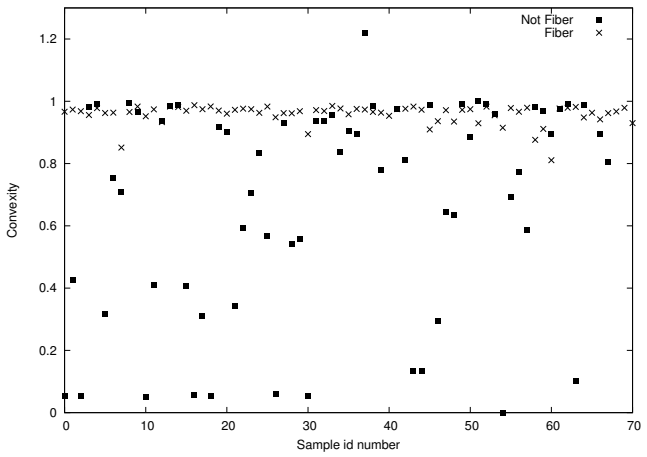


(a) Eccentricity

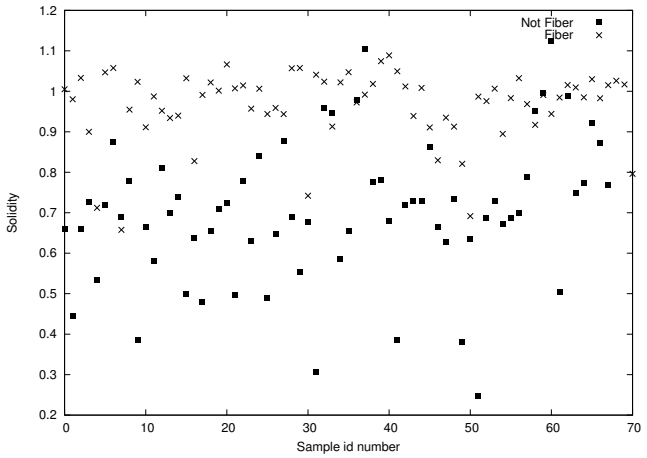


(b) Size

Figure 5.6: Plots of size and eccentricity values of regions



(a) Convexity



(b) Solidity

Figure 5.7: Plots of solidity and convexity values of regions

Descriptor	Lower threshold	Upper threshold
Eccentricity	1.0	50
Convexity	0.8	1.0
Solidity	0.60	1.2
Size	$82\mu m^2$	$8200\mu m^2$
Circularity	0.5	4.0

a. Slack thresholds

Descriptor	Lower threshold	Upper threshold
Eccentricity	1.0	40
Convexity	0.85	1.0
Solidity	0.9	1.1
Size	$82\mu m^2$	$8200\mu m^2$
Circularity	0.5	2.0

b. Strict thresholds

Table 5.1: Thresholds for lumen detection by descriptors of lumen shape

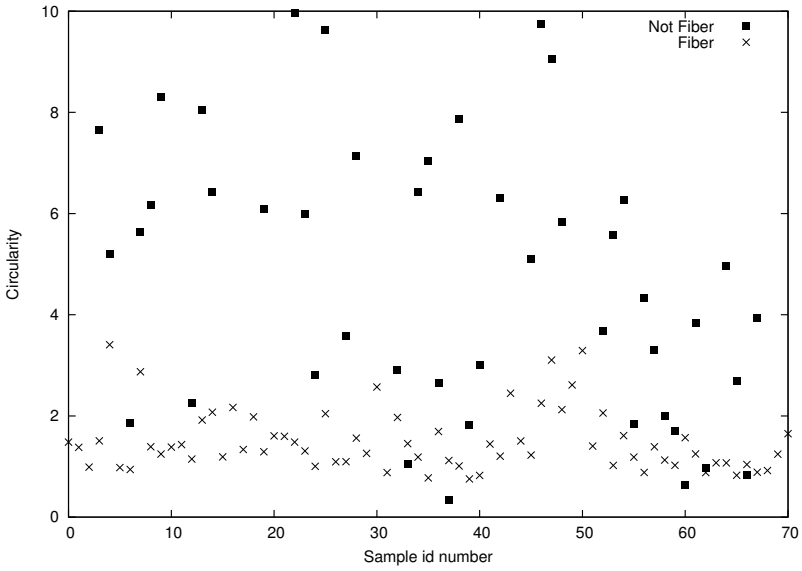


Figure 5.8: Plot of circularity values of regions

segmentation framework, FibForsk.

We have created an algorithm that uses the detected lumen candidates as input. Effort has gone into making the algorithm as simple as possible. The traced lumens are later re-evaluated according to their size and skeletal properties, making it possible to throw away clearly falsely identified lumens.

1. For each slice, evaluate all disjoint regions, and seed suitable candidates (see Section 5.2.4).
2. For all seeds that does not yet point to a label:
   (a) Trace fiber from seed point in both directions (see Section 5.2.4)
   (b) Join regions where the following conditions are met:
      - The regions are 6-connected
      - The average distance between all coinciding slices is < maxDist
   (c) Join regions of they are connected on endpoints along the same axis of orientation.
   (d) Remove region if length of fiber < minLength

(a) refers to the tracking algorithm presented in Section 5.2.1.

(b) ensures that split lumens resulting from a collapsed fiber is automatically merged into the same region, assuming that they are connected in at least one voxel.

(c) attempts to automatically joins lumen-regions that are aligned on the same path, indicating that they belong to the same fiber but are untraceable in a single point along that path. In some cases this can also merge lumen regions that were traced along different axis, indicating a severely bent lumen.

(d) rejects lumens that cannot be traced for more than a couple of slices, as they are likely not correctly identified. However in some cases such regions are parts of a fiber. Rather than deleting the label permanently it is given a special label, such that it can be merged using the criteria stated in (b) or (c), but is deleted automatically after the entire process has completed.

5.3 Discussion

We have implemented a practical tool for segmentation of fibrous materials. Our fiber-tracking procedure implements some of the techniques suggested for tracking lumens in (Holen and Hagen 2004), but we have extended it to

provide a full segmentation along all axis of orientation and a fully three-dimensional wall segmentation that removes directional bias from the data. An effort was made to keep the method as simple as possible.

A method for a fully automatic segmentation that can reduce the time required to segment a volume considerably was invented. Manual inspection and correction is required after the automatic lumen-segmentation is performed, and tools for these tasks have been provided. The performance of the auto-segmentation is data-dependent. We estimate the ratio of the number of correctly segmented lumens to the total number of lumens in the final segmentation to be 50–70% in our experiments. The difference in speed between the HH–approach and this one is significant. While Runar Holen reports 3–4 weeks of combined manual interaction and processing time were used to segment one volume (S8), we found that segmenting the same volume with our tools took about 3 hours and one night. The night was used to run automatic lumen segmentation and the 3 hours were used to fine tune the resulting segmentation and run wall-segmentation.

We found the convexity of an area to be a good basis for deciding if that area is a lumen or not. Rejecting areas with a high eccentricity value as lumens also improved the automatic detection of lumens significantly. The curvature, used by Aronsson (2002) and Lunden (2002) for lumen detection, did not distinguish significantly between lumens and non-lumens in our experiments, although there was a slight tendency for lumens to have higher curvature values than non-lumens.

The rules used for thresholding should be examined further. The relations between descriptor values is of special importance, rules combined in a special way can have more success than our simple step-wise, linear, min-max rules. We believe this issue can be addressed by using automatic rule learning algorithms and can result in improved rules and new domain knowledge from these rules. A larger data-set should be input to a decision learning algorithm, like C5.4 (Quinlan 1993), for example in the Weka[1] machine learning program. A part of the data-set should be used as a verification set, to test the validity of the resulting rules.

[1] `http://www.cs.waikato.ac.nz/~ml/`

Chapter 6

Measuring the orientation distribution

The mechanical properties of fiber-reinforced composites are largely determined by four primary factors: fiber content, aspect ratio, orientation, and the matrix itself. In particular, fiber orientation plays a very important role (G. Zak and Benhabib 2001). This chapter presents a method for measuring fiber orientation, as well as measurements on a selection of material samples.

6.1 Previous efforts

In Yang (2001, p.143), PCA was applied to the segmented voxel structures in order to measure the orientation distribution of fibers. Rather than plotting the eigenvectors as straight lines with origin at $(0, 0)$ circular diagrams were used, indicating each vector by its end-point. This was done to make the plot visually clearer. The vectors were normalized and defined to have a positive z-coordinate. Separate plots were made for all three principal components. The following analysis was based on visual inspection of these plots.

6.2 Methods of measurements

While the focus in this thesis have been to segment and identify individual fibers, we have included some bulk measurements as well, as they are descriptive of material properties. We will focus on the fiber orientation distribution.

6.2.1 Bulk measurements

This section presents the measurements we will do on the binarized (bulk) volumes.

Porosity and volume fraction. *Porosity* is a commonly used measure for describing porous materials. Consider a sample of total volume V. Define the volume of the solid phase to be V_s, and the volume of the pore phase (the holes) to be V_p, with $V = V_s + V_p$. The volume fraction is a normalized variable that is generally more useful. The volume fraction of the pore phase is commonly called the porosity, and is denoted $\Phi = \frac{V_p}{V}$. The solid volume fraction is then $1 - \Phi$ (Garboczi et al. 1999).

Mean fiber chord measurements. The *Equivalent pore theory* is a concept for obtaining 3D characteristics of paper from cross-sectional images. It is a statistic, in that it gives a representation of the homogenized three-dimensional porous structure as an ellipsoid (Holmstad 2004, p.41). This principle can be applied to both mean pore chords and mean fiber chords.

We have calculated the mean fiber chord measurements of the bulk material. They are estimated by the average measured length of continuous non-zero voxels along each of the three primary axis relative to the viewing plane. A more advanced method based on measuring the mean pore chords for 23 different spatial directions is presented in Holmstad (2004).

6.2.2 Fiber orientation distribution

There are many ways to define the orientation of a fiber. The simplest method is perhaps to use the end points of each fiber or its skeleton to define its orientation, however in some cases defining the skeleton and end-points is a difficult or time-consuming task.

We have used PCA on three-dimensional data to calculate the fiber orientation distribution in a completely domain-independent way. As input to PCA we used fibers from the individually segmented volume. A direction vector was calculated for each fiber. Creating statistics on directions is complicated by the direction duality problem. We will calculate the orientation vectors and provide statistics on the fiber orientation distributions.

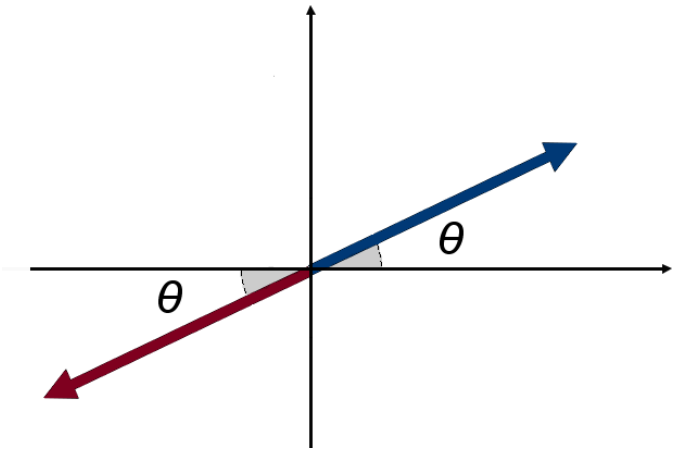


Figure 6.1: Illustration on how two inverse vectors can describe the same orientation.

The orientation duality problem

The orientation of each fiber indicated by a vector can result in two different vectors pointing in exact opposite directions, both valid orientations for the same fiber (figure 6.1). While both are still valid directions when calculating the mean value, we may risk that two exactly opposite vectors cancel each other out in the average measurements even though they represent the same orientation. To compensate for this effect we can define the z-coordinate to be positive (Yang 2001). This is done by changing the sign of all components of vectors that have a negative z-component. Note that the selection of the z-plane was arbitrary, and will result in that the mean vector always has a large z-component. In the unlikely case that two fiber-vectors lie exactly in the z-plane, we can apply a similar technique by requiring the x-coordinate to be positive.

Statistics on the fiber orientation distribution

On the resulting vector field, we calculate the normalized average vector of the field. This average vector represents the average orientation of the entire fiber volume. The average is not descriptive if the deviation is very large, so we have calculated the component-wise variance as well. In addition, we have calculated the average deviation angle between all vectors and the average vector by using the average angle of the dot product of each fiber orientation vector and the average orientation vector. A low average angle will indicate that all fibers are aligned in mostly the same direction. The average deviation angle of one volume can be compared to the average deviation angle of others to compare fiber alignment.

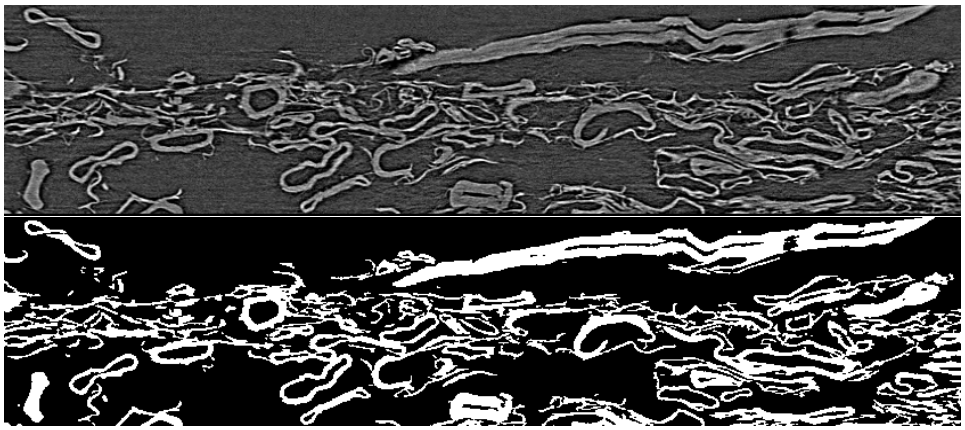


Figure 6.2: Sample from original S8 (top) and sample from binarized S8

6.3 Different Materials

We have binarized a series of volumes using the method presented in Chapter 4. A description of these volumes follows.

S8: TMP-based newsprint, freeness 145 ml, shive content 5.5 shives per gram of pulp, not calendered, density 387 kg/m^3. Paper.

A sample from the volume and the binarized version of that sample are shown in figure 6.2.

STFI7: Unconsolidated fiber mat. Fiber mat = Lab made softwood kraft fiber, kappa = 46, 2x 0.5 bar pressure, i-PrOH solvent exchange, Matrix = epoxy vinyl ester, weight fraction = 20.1%. Composite.

A sample from the volume and the binarized version of that sample are shown in figure 6.3.

STFI9: Consolidated fiber mat, Fiber mat = Lab made softwood kraft fiber, kappa=46, 0 bar pressure, Matrix = epoxy vinyl ester, weight fraction = 20.4%. Composite.

A sample from the volume and the binarized version of that sample are shown in figure 6.4.

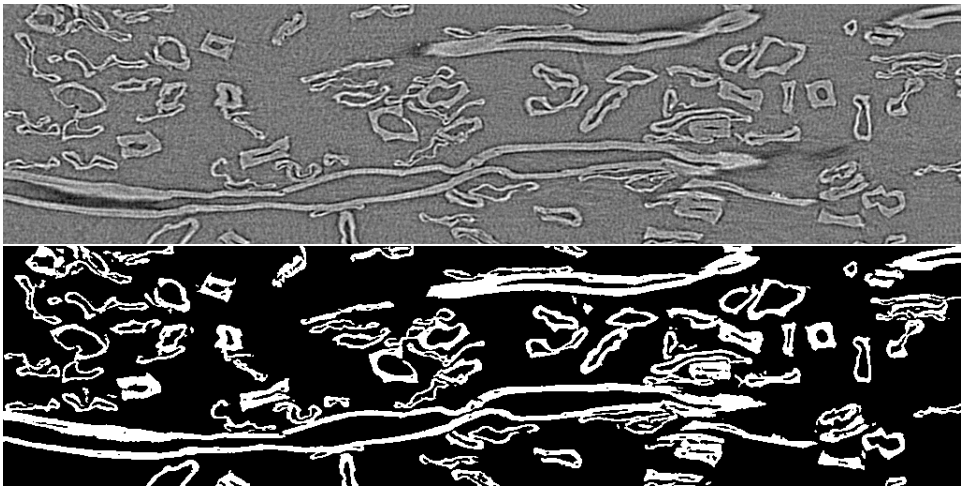


Figure 6.3: Sample from original STFI7 (top) and sample from binarized STFI7

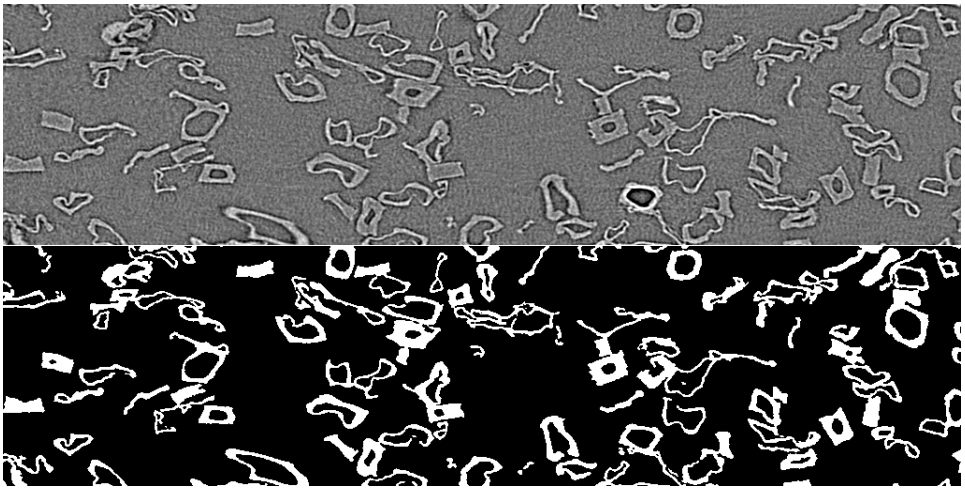


Figure 6.4: Sample from original STFI9 (top) and sample from binarized STFI9

6.4 Measurements

This section presents the results of measurements we have done on segmented volumes.

6.4.1 Volume S8

This section presents the measurements we have done on the S8 volume.

The volume was first first segmented using the HH–approach (Holen and Hagen 2004). Then we performed a full segmentation using our own approach. We did this both to compare measurements on the two differently segmented volumes and to compare the segmentation methods used.

The bulk measurements are approximations only, as the volume contained parts of a large capillary tube in addition to the paper sample. The volume was cropped so that the capillary was removed from the entire volume. Some fiber material close to the capillary was also removed in this process.

Sample size in pixels : $1215 \times 435 \times 1100$

Bulk-level measurements

Note that rather than using the entire volume, we cropped the material so that the measured region was inside the paper material, thus avoiding having the problem of having to define the paper surface. Using a rolling ball algorithm for defining the surface was regarded as a more correct method by Holmstad (2004).

Material density : 29.96%
Average length across fiber phase : $x = 9.30, y = 4.16, z = 11.23$
Average length across pore phase : $x = 22.82, y = 10.20, z = 27.54$

Fiber-level measurements using the HH-approach

The individual segmentation of S8 was used for fiber-level measurements (figure 6.6). The resulting orientation distribution is shown in figure 6.5.

Segmented fiber count : 171
Percentage of identified material : 30.92 %
Average normalized orientation vector: $< 0.05, -0.07, 0.99 >$
Standard deviation from average: $x = 0.51, y = 0.17, z = 0.26$
Average deviation angle from the normalized orientation vector: 29.96°

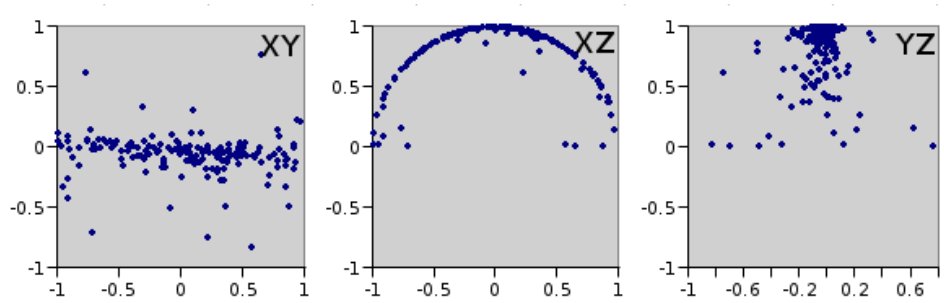


Figure 6.5: Orientation distribution plots of XY, XZ and YZ axis of HH–segmented S8

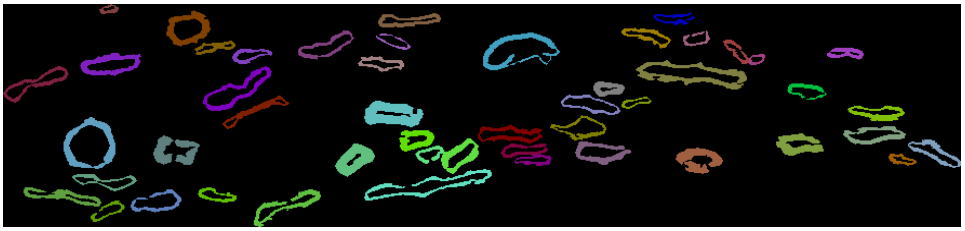


Figure 6.6: Example of HH–segmented data from the S8 sample

Fiber-level measurments using the improved fiber segmentation

The individual segmentation of S8 was used for fiber-level measurements (figure 6.8). The resulting orientation distribution is shown in figure 6.7.

Segmented fiber count : 220
Percentage of identified material : 51.20 %
Average normalized orientation vector: $< 0.13, -0.06, 0.99 >$
Standard deviation from average: $x = 0.56 y = 0.14 z = 0.26$
Average deviation angle from the normalized orientation vector: 33.34°

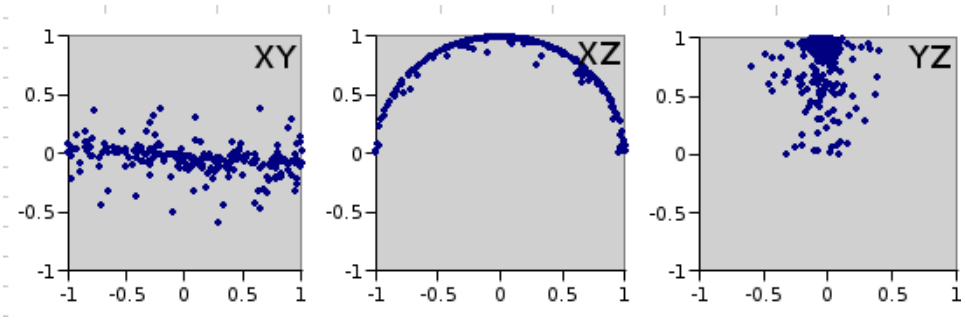


Figure 6.7: Orientation distribution plots of XY, XZ and YZ axis of s8

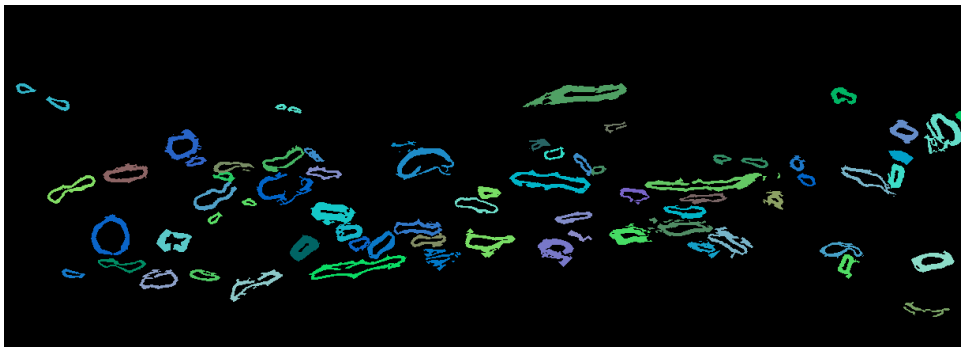


Figure 6.8: Example of segmented fiberdata segmented from s8 using our improved method

6.4.2 Volume STFI7

This section presents the measurements we have done on the STFI7 volume.

Sample size in pixels : $1024 \times 256 \times 1024$

Bulk-level measurements

Material density : 16.82%
Average length across fiber phase $x = 12.97, y = 5.18, z = 9.12$ Average length across pore phase $x = 64.08, y = 25.62, z = 45.06$

Fiber-level measurements

The individual segmentation of STFI7 was used for fiber-level measurements (figure 6.11). A ray-traced rendering of the same segmentation data is shown in figure 6.9. The resulting orientation distribution is shown in figure 6.10.

Due to a high amount of noise and severely deformed lumens, the automatic tracking was only able to identify about 23.5% of the material in this sample, and this volume had to be segmented in a fairly manual fashion.

Segmented fiber count : 179
Percentage of identified material : 64.42%
Average normalized orientation vector: $< 0.32, -0.07, 0.94 >$
Standard deviation from average: $x = 0.51 y = 0.17 z = 0.56$
Average deviation angle from the normalized orientation vector: 29.9°

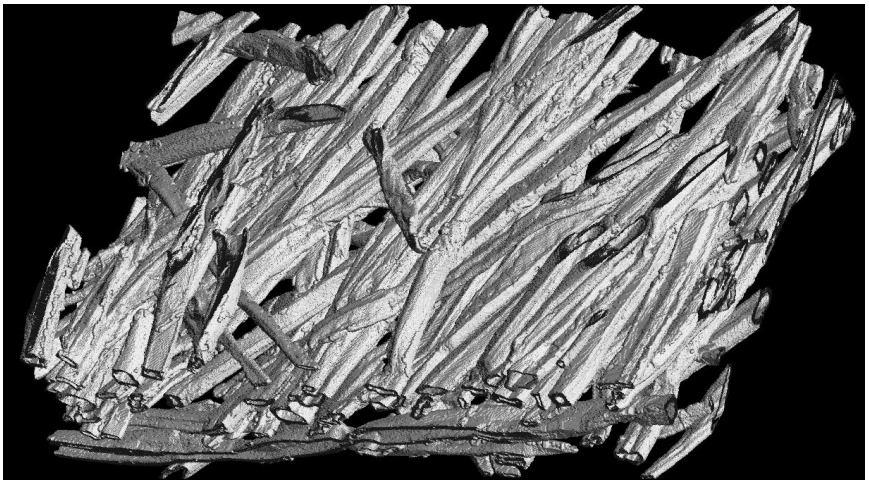


Figure 6.9: Ray-traced rendering of the segmented STFI7

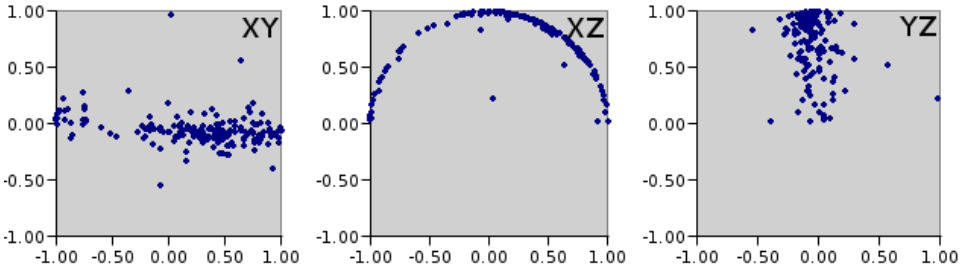


Figure 6.10: Orientation distribution plots of XY, XZ and YZ axis of STFI7

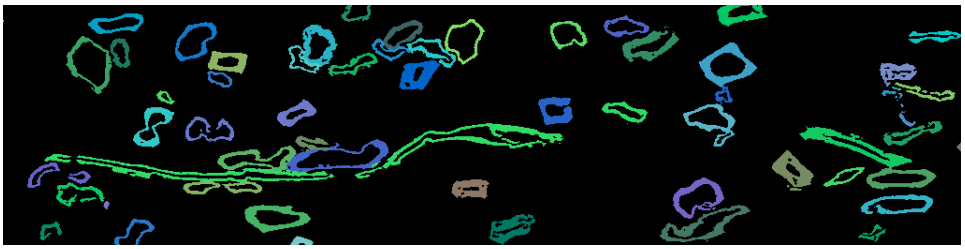


Figure 6.11: Example of segmented data from the STFI7 sample

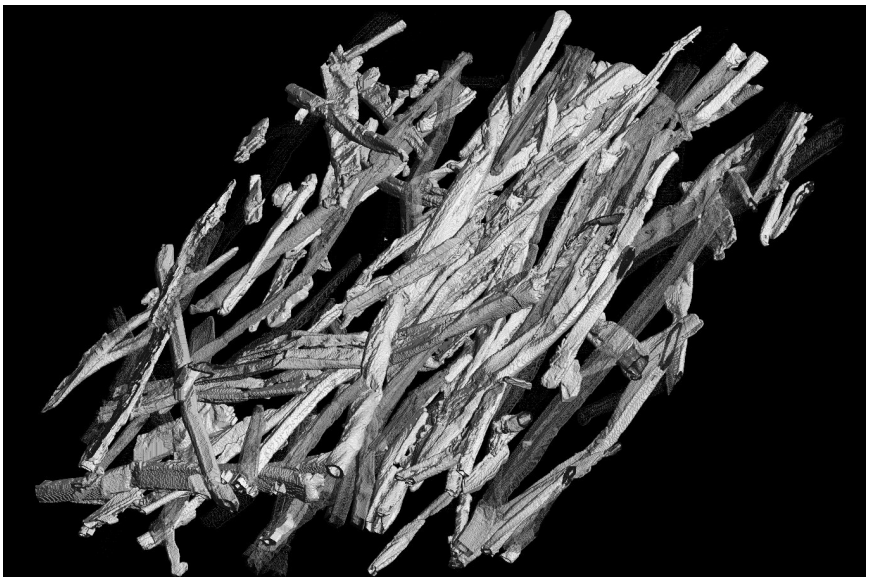


Figure 6.12: Ray-traced rendering of the segmented STFI9

6.4.3 Volume STFI9

This section presents the measurements we have done on the STFI9 volume.

Sample size in pixels : $1024 \times 256 \times 1024$

Bulk-level measurements

Material density : 17.3%
Average length across fiber phase : $x = 16.59, y = 11.38, z = 30.57$
Average length across pore phase : $x = 51.30, y = 35.20, z = 94.53$

Fiber-level measurements

Corrections were made by manual inspection after auto segmentation was applied, and regions having a size of less than 500 voxels were rejected. 3D wall segmentation by geodesic distance transform was applied, allowing for a thickness of up to 20 voxels. A rendering of the resulting volume, used for measurements is shown in figure 6.12.

The individual segmentation of STFI9 was used for fiber-level measurements (figure 6.14). The resulting orientation distribution is shown in figure 6.13.

Segmented fiber count : 239 Percentage of identified material : 59.8%
Average normalized orientation vector: $< 0.049, 0.010, 0.998 >$

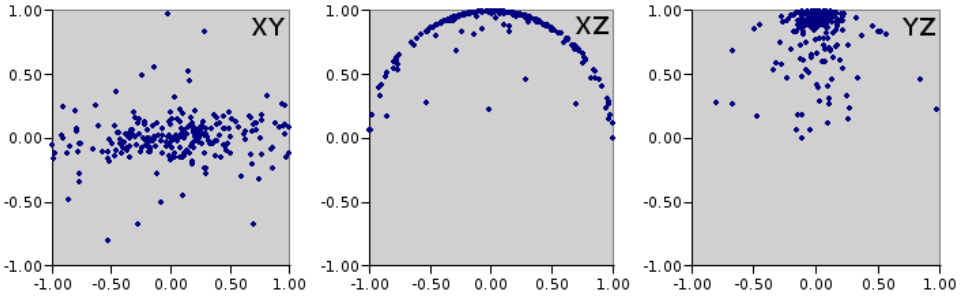


Figure 6.13: Orientation distribution plots of XY, XZ and YZ axis of STFI9

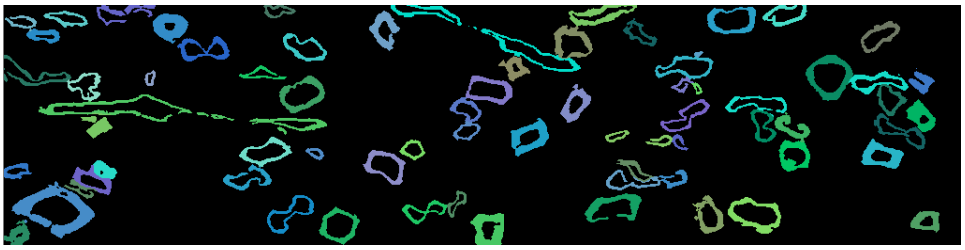


Figure 6.14: Example of segmented data from the STFI9 sample

Standard deviation from average orientation: $x = 0.46y = 0.19z = 0.22$
Average deviation angle from the normalized orientation vector: 26.42°

6.5 Discussion

We segmented the S8 volume to demonstrate that our proposed method is applicable to paper fiber segmentation, as well as the segmentation of fiber-reinforced composites. The segmentation was also used to compare the new method with the method of Holen and Hagen (2004). This volume is difficult to segment accurately, but we were able to pick up significantly more material than Holen and Hagen (2004). This is mainly because we segmented lumen from all axes of orientation, but also because our FibForsk software made it easier to manually inspect the volume. The fibers in this sample were aligned fairly close to straigt line through the voluem, but the extra cross-directional fibers we added in our segmentation increased the angle of deviation slightly (from 30% to 33 %). In addition, the x-component of the mean orientation vector increased, indicating that the added fibers are indeed cross-directional.

The STFI7 and STFI9 volumes are interesting to compare because they show the same type of material, but more pressure was applied to the former in the production process. We believe this is reflected in the orientation dis-

tribution plots, as the STFI7 volume is more compressed in the XY-range, indicating that fiber orientation was affected by pressure applied from the top of the volume during production. Note that because the sample may have been oriented differently when imaged than when produced, the direction of pressure can be from any one of the six sides of the sample in the production process. In addition, the deviation angles and standard deviations from the mean orientation vector indicate that fibers are somewhat more aligned in the same direction in the STFI9 sample.

The basic mean pore- and fiber- chord measurements does not seem to correlate well with the fiber orientation. If the pore chord measurements are to be used as indications of fiber orientation, more directions than the three primary axes are probably needed.

Chapter 7

Implementation

The primary goal for our implementation has been to develop and implement a practical tool to aid the analysis of three-dimensional images of fibrous materials.

In Holen and Hagen (2004) a method individual segmentation of fiber images was suggested. Our goal has been to improve and continue this work.

Different fibrous materials tend to have widely different characteristics, and most algorithms fail to cover all cases. To make these tools practical for future use, emphasis has been put on expandability, re-usability and configurable solutions.

7.1 User guide

This section aims to give new users a short introduction to the FibForsk application.

7.1.1 Introduction

The FibForsk application is an interactive tool as well as a programming framework to aid material scientists in the segmentation and analysis of three-dimensional volumes. It is primarily made for wood fiber segmentation, but efforts have been made to keep the base architecture as open, modular and reusable as possible to enable expansion and experimentation in other areas as well.

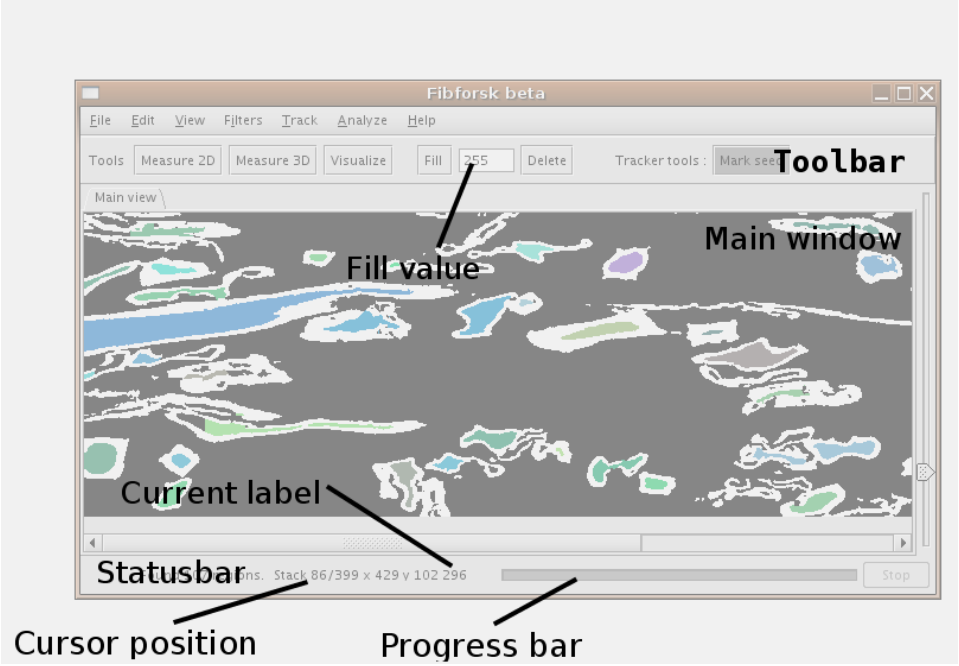


Figure 7.1: Overview of the FibForsk interface

7.1.2 System requirements

1. Java 1.5.0 or later (also called Java 5.0)
2. 256 MB main memory

7.1.3 Licensing

The FibForsk application is in the free domain.

7.1.4 Installation

Installation is simply a matter of copying the FibForsk application folder to the primary hard drive. Edit the run file (or Makefile on Unix/Mac OS) to reflect the maximum amount of available memory on your system by altering the -Xmx parameter.

7.1.5 User interface

Figure 7.1 shows the main components of the GUI interface. The *Toolbar* is contains a series of toggle-buttons, indicating the currently selected tools. The currently available tools are :

- Measure 2D : Applies and prints results from all available 2D measurement descriptors Section 7.2.8 to a selected -connected 2D region
- Measure 3D : Applies and prints results from all available 3D measurement descriptors to a selected labeled region
- Visualize : Opens a visualization of the selected labeled region in a separate visualization window Section 7.2.6
- Fill : Applies 26-connected three-dimensional flood-fill to the selected voxel using the indicated *Fill value*.
- Delete : Applies 26-connected three-dimensional flood-fill with the label 0, effectively deleting the region.

Note that other tools may be added or removed depending on the currently selected Tracker-plugin.

The *Main Window* contains a 2D window into the volume. The user may scroll through the volume by using the *depth slider* on the right hand side of the main window.

The *Status bar* indicates the cursor's current x,y,z position, as well as the value/label of the voxel directly below the cursor cross-hair. The *Progress bar* indicates the status of the current process. When a process is running, the *Stop* button may be pressed to terminate it. Note that this is not equivalent to canceling the action, and pressing this button can leave the program in an unstable state.

7.1.6 Performing individual fiber segmentation

The work flow for segmenting a binary volume in FibForsk consists of a series of steps required to successfully segment the volume.

The volume is first binarized as explained in Chapter 4. Since this procedure is highly data dependent, we have not included this functionality in the FibForsk application.

This work flow describes the main steps involved in correctly segmenting a fibrous volume using the FibForsk tool and the algorithm described in Chapter 5.

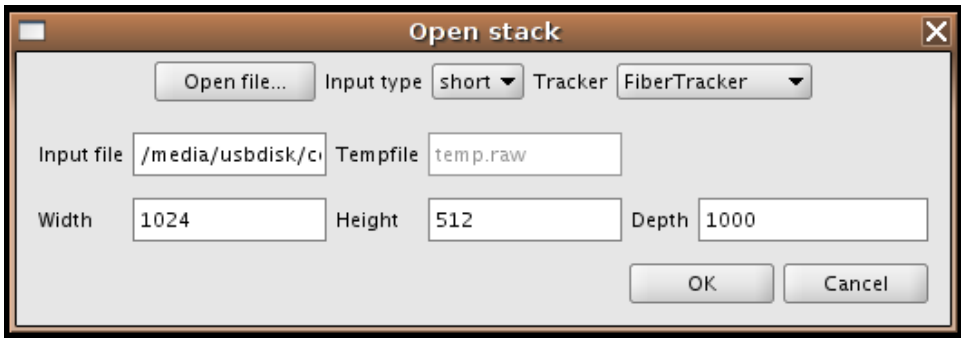


Figure 7.2: The opening dialog in FibForsk

Starting up

In Linux, the FibForsk application can be started by calling the *run* command on the command line. In Windows this can be performed by clicking the ”run(.bat)” application icon.

It is important that enough free memory is available to FibForsk. By default all slices will be loaded into the primary memory. If not enough memory is available, extensive disk-swapping will make most operations inefficient. However, it is possible to enable a special Slice-based cache that will allow the fiber-tracking to execute efficiently even though the host computer cannot hold more than a few slices in memory at once. To do this, open the preferences panel from the *edit*-menu and set the *use_cached_access* property to 1. The user may safely leave the temp-file settings with the default value, unless there is little storage space left on the host hard drive.

Workflow

1. The open dialog shown in figure 7.2 will automatically open upon start, and can be also be reopened with *File→Open*. The user must indicate the correct data type, width, height and depth of the binarized volume. Note that all previously saved volumes will always be in short-format. He must also ensure that the *FiberTracker* is selected from the tracker selection menu.

2. The user may choose to do the segmentation manually, however it is recommended that *auto segmentation* is selected from the menu first. This uses the method suggested in Chapter 5, to pick up the best candidates in evenly spaced intervals along three spatial directions separately. Depending on the quality of the binarized volume this will often detect most of the candidates.

3. After the auto tracking is complete, the user inspects the volume using the depth slider on the right hand side, and seeds undetected fibers if necessary using the *Mark seed*-tool on the toolbar. Erroneously detected lumens can be removed from the 3D volume by using the delete-tool available on the tool bar, and regions can be joined by using the Fill-tool on the toolbar. Right-clicking a region automatically sets the fill-value to the label of the clicked region. When enough seeds have been selected, the *Track*→*Track lumens*-option initiates tracking in both directions for each of the seeded lumens. The seed-list can be cleared by selecting *Track*→*Clear seeds* as well.

4. Flip the volume along the two other principal axis and repeat the previous step until no more lumens can be found.

5. After all traceable lumens have been isolated, the final segmentation-step can be applied by selecting *Track*→*Track walls 3D* from the menu.

6. Save the final segmented volume by selecting *File*→*Save*.

7.2 Architecture

This section aims to give a brief overview on the design goals and implementation details of the FibForsk application.

7.2.1 Requirements

The purpose of this application is to be a practical tool for fiber segmentation and analysis. The tool should be able to analyze, segment and present usable statistics on binary data. This section will present the requirements we used as a basis for our design.

Primary requirements

1. Binarization of fibrous data
2. Automatic detection of individual paper fibers
3. Segmentation of individual fibers
4. Analysis of this data

Secondary requirements

1. The software should be able to load raw binary data in short- and byte-format.
2. Segmented volumes must be loadable by the ImageJ-platform.
3. The tracker-components must be interchangeable and separated from the user-interface
4. The paper tracker must support data analysis of segmented fibers including estimates for
   (a) Fiber length
   (b) Fiber size in voxels
   (c) Fiber surface area
   (d) Fiber orientation
5. The software must be able to run on both Linux- and Windows platforms, having a minimum of 512 MB primary memory.
6. The software should have a modular architecture, allowing software to be reused and developed to support different segmentation algorithms.

7.2.2 Development platform

The FibForsk application was developed in Java to ensure portability and compatibility with existing software in use at PFI. Certain key components have been programmed in C, and integrated using the Java Native Interface (JNI) to optimize performance-critical components.

The main development platform was Linux Ubuntu 5.04 using Sun Microsystems Java 1.5.02. All components have been successfully tested on Microsoft Windows XP Professional and Mac OS 10.4 (Tiger).

7.2.3 Base architecture

FibForsk is a stand-alone application with a graphical user interface. While it was initially intended to be an extension to the public domain ImageJ image analysis environment, this did not give us enough flexibility to experiment with user interaction in the process of tracking fibers. Another important reason for not using ImageJ as the host environment was the need to use a more flexible storage and data management, than what was

possible with ImageJ. Loading data sets using ImageJ requires the entire volume to be resident in main memory at once. This places strict requirements on primary memory (Section 7.2.5) as well as restrictions on the maximum size of the volumes to be analyzed.

FibForsk is linked to libraries provided by ImageJ (Rasband 1997-2005) as well as Jama (team 1999-2005). All libraries used are in the public domain.

7.2.4 Package and class-overview

FibForsk is the name of the application we have developed, to perform fiber segmentation as well as data analysis.

The FibForsk code base is divided between the main packages core, descriptors and trackers. The core-package consists of most of the core functionality of FibForsk, including the user-interface (core.gui), IO (core.io) architecture and reusable image processing libraries (core.Imagefunctions, core.Watershed etc). A simplified UML class-diagram is presented in figure 7.3.

7.2.5 Storage

This section presents a description on the file-formats and methods for file access used in the FibForsk application.

File format and output

Save-data is exported to short-valued little-endian raw-format, appending an empty byte when loading and saving single byte raw volumes.

We have reserved the range 0–255 to indicate gray-level information in the data, and such values will be indicated by gray-levels within the FibForsk application. These do not indicate segmented regions, and are preserved when saving a data volume to file. Their primary use is for loading original gray-scale data or distance maps as input to the segmentation algorithms. This implies that loading byte-based images and saving them, will always result in an equivalent short-valued image. By default this will appear as a dark gray-scale image when exported to other image-applications, since we have effectively only padded an extra byte to each byte in the input.

Any values from 256 and up to 8192 may be used for labeling, giving a maximum of 7196 separate regions/fibers by default. The maximum fiber count is currently 4096, but this number can be increased by adjusting the *maxFiberValue* property in the properties dialog at the expense of slightly

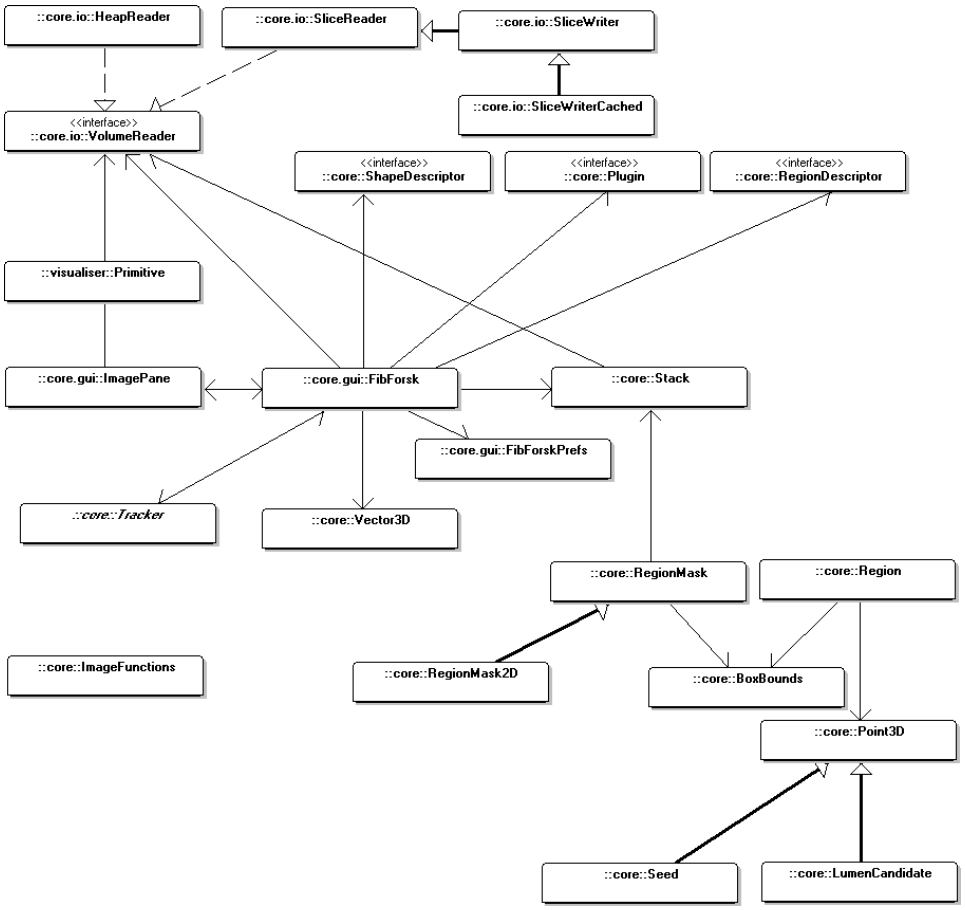


Figure 7.3: UML class diagram for the FibForsk application

higher memory requirements when performing measurements on segmented regions.

This leaves the three higher-order bits reserved to serve as internal temporary flags by any tracking routines.

File input

The FibForsk application can import data in RAW byte- or short-format. Each voxel is packed in a row-by-row, slice-by-slice order. When short-data is read, the lower-order byte comes first (little endian).

File management and cache

For segmentation purposes we require fast read- and write-access to data on a pixel-by-pixel basis. This makes most compression-schemes unfeasible. In addition, all segmented regions require a separate label. Since the amount of fibers in a typical paper volume easily exceeds 256, the maximum value available within a single byte, we use two bytes per voxel for all data-storage internally.

All data access is handled by the *VolumeReader* interface. We have two different implementations of this interface, namely *SliceReader* and *HeapReader*. Both give the same basic functionality, but can be interchanged depending on the type of data access required and the amount of available memory on the host computer.

The SliceReader caches separate image-slices in a FIFO buffer, not requiring the entire data structure to remain in memory at once, effectively moving a slicing window of data across the volume. Since our fiber-tracking routine operates in a linear fashion through the volume this form of cache can be efficient.

The HeapReader implementation, works by loading all slices into heap memory, or on disk using any built-in swapping algorithms provided by the host operating system. This latter method is preferred if the host platform has enough primary memory to support it, and is used by default. To change this behavior, set the preference *use_cached_access* to *1*, under the *edit→preferences* menu.

The Stack class is another abstraction layer to the data, giving the user access to individual voxels, size, orientation and other information regarding the data.

Caching

Efficient caching is essential to avoid performance loss, as access to secondary storage is very slow. Some caching is already provided by the hardware, as well as the operating system. However, these caching algorithms do not employ any knowledge about the data we are accessing and we have therefore developed a simple, yet effective cache-scheme optimized for our particular tracking-method.

The type of cache that we have implemented is based on the fact that our primary tracking method generally requires a slice-by-slice access. Access to individual imageSlices, has therefore been optimized. We have developed a set of classes, SliceReader and SliceWriter enabling us to randomly access 2D-slices from a 3D volume on disk using a moderately large LIFO-based write-back cache.

This implies that changes to a cached slice, will *not* be reflected on disk unless the particular slice is replaced in cache or the writer is explicitly flushed. All algorithms employing the SliceWriter class should therefore be written to ensure that such caches are flushed explicitly after completion, by calling the *<SliceWriter>.flush()*-method to ensure that results are properly stored to disk.

The actual replacement policy can easily be modified by overloading the method updateCache() in the *SliceWriter*-class. The amount of slices to store in cache is configurable by changing value of the *defaultSliceCount*-property from the preference-panel on the edit-menu.

7.2.6 Data visualization

We have integrated the FibForsk application with a visualization module written by Terje Tubaas (Tubaas 2005). This enables the user to rotate and view segmented regions in three-dimensions using an iso-surface representation based on the marching cubes algorithm, and can be started by selecting the Visualization tool from the toolbar, and clicking on the region to be inspected (figure 7.4). Note that the plugin currently has very high memory requirements and is only available on the Windows-platform.

Usage

The inspected volume rotates around the y-axis. It can be moved around in the x-y plane by holding down control while dragging the mouse. Zoom can be controlled in a similar manner by holding down the shift button. An option-menu is available by right-clicking in the visualization window.

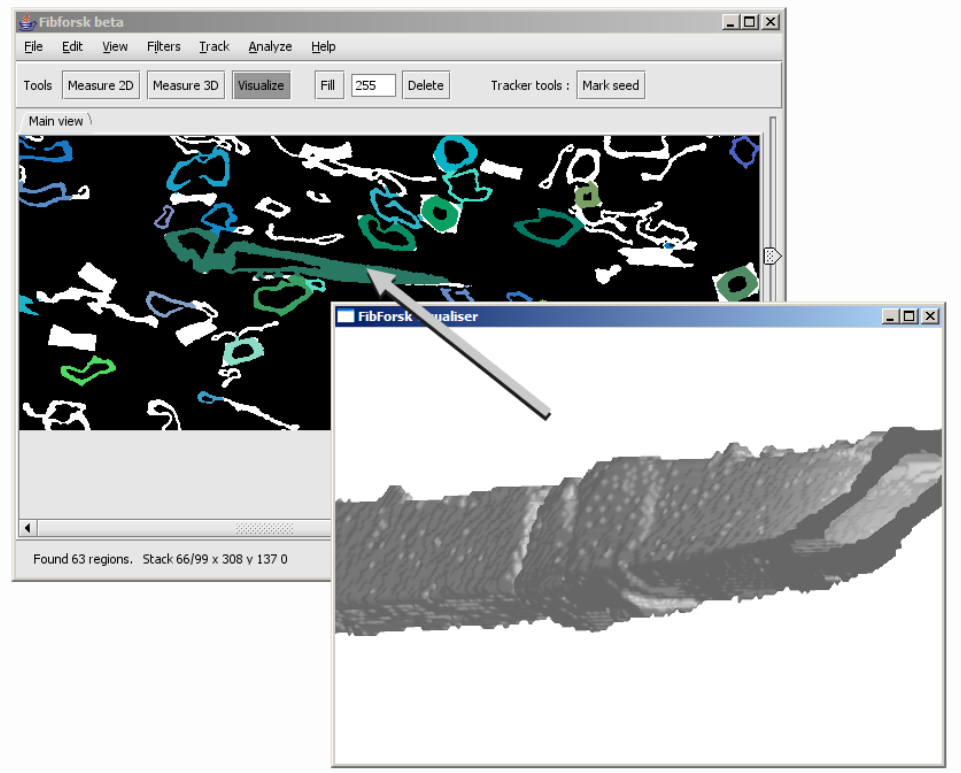


Figure 7.4: Example visualization of the fiber indicated by the arrow.

7.2.7 Trackers

The tracker is the class responsible for the actual segmentation. Any segmentation routine depending on domain knowledge will have to be altered when applied to a different type of data, even though a lot of the functionality required is similar. When trying out different approaches on the same data it is also practical to be able to compare the performance of different tracker-methods. Thus we have made the tracker a pluggable interface in the FibForsk framework.

The tracker has access to the volume-data, the user libraries provided by FibForsk, as well as those provided by ImageJ.

For a complete description of the Tracker-interface, consult the source code of the file core/Tracker.java.

Implementing and installing a new tracker

A new tracker-class can be created by implementing the Tracker interface located in the core-package. The resulting class-file must then be placed in the *trackers* package. Upon restart of the FibForsk application, the new tracker will automatically be detected and available for selection in the Open-dialog.

Storage of volume-data output during tracking

When tracking fibers, it is often useful to create and compare several data volumes as output of any given algorithm. This makes it possible to debug graphically what is happening at several different stages, as well as storing temporary structures that can be re-used for other purposes. We have developed a programming-interface to present the contents of a particular SliceWriter to the main-window in FibForsk, resulting in a separate tab in the main window. This makes it easy to switch back and forth between the separate output volumes for comparison. All these windows must have the same dimensionality as the main volume.

Example usage This particular example shows how to connect the output of a particular SliceWriter to the main application window, so that the user can select it while browsing through the volume and compare it to the other visible data-channels.

```
SliceWriter lumenlog = new SliceWriter("lumens.raw", width,
height);
FibForsk.getInstance().addChannel(lumenlog, "Lumens");
```

7.2.8 Descriptors

Analyzing a segmented volume is done by applying a set of measurements, such as size, orientation, curvature etc. These do not change the volume in question but usually return numerical data on different properties. We have introduced a particular interface for exactly this purpose, making it possible to easily add new measurements to the already implemented modules by implementing only the relevant descriptor-module.

Descriptors designed to work on three-dimensional data can be recognized by exploiting the fact that they implement the method does3D(). If a descriptor returns a single double-typed value, they implement the isNumeric() method. The latter makes the descriptor available to any function assuming a single-valued input, such as the automatic seeding algorithm presented in Chapter 5;

An *ArrayList* list containing all currently available descriptors can be obtained by calling the method *FibForsk.getDescriptors()*.

```
package core;
public interface RegionDescriptor {
    public Object measure(RegionMask mask); //Generates the
        descriptor result
    public String getName();  //Returns the name of this
        descriptor
    public String getAbout(); //Returns a short description of
        this descriptor
    public boolean does3D(); //Returns true if this descriptor
        is applicable on 3D data
    public boolean isNumeric(); /*Returns true if the measure
        returns a
        Double-object as its only result */
}
```

Implementing and installing a new 3D region-descriptor

Region descriptors can be used to generate statistics on segmented 3D data. Examples of descriptors are included in the *descriptors* package.

A new descriptor-class can be created by implementing the *RegionDescriptor* interface located in the core-package. To be available for 3D region measurements the method does3D() must be overloaded to return *true*. The resulting class-file must then be placed in the *descriptors* package. Upon restart of the FibForsk application, the new tracker will automatically be detected and made available for selection in the *Analyze→Region measurements* menu.

Using descriptors

When the descriptor is selected from the *Region measurements menu*, the data will be analyzed by creating a set of box-bounded *RegionMask* objects that are passed to the descriptor's measure()-method. The result for each labeled region will be listed in the console window of the FibForsk application.

7.2.9 Image functions

The *ImageFunctions* class contains java implementations of many of the algorithms presented in Chapter 3. Some of these methods are:

- 2D- and 3D flood-fill and masked flood-fill
- 2D- and 3D dilation, erosion and masked (geodesic) dilation
- 2D- and 3D distance maps
- 3D geodesic distance maps
- linear interpolation methods

7.3 Tracking Fibers

The FiberTracker class is an implementation of the Tracker interface, and represents an implementation of the ideas presented in Chapter 5. We have included a selective source code listing in Section B.3. Only the components that directly affect paper fiber segmentation have been included. Complete source code will be made available online or by direct request to the authors.

7.4 FibForsk

FibForsk is available for free download from `http://www.pvv.org/~perchrh/fibforsk`.

7.5 ImageJ-plugins

We have made a number of plugins to ImageJ (Rasband 1997-2005) as a part of this thesis.

ImageJ is a public domain Java image processing program inspired by NIH Image for the Macintosh. It runs either as an online applet or as a downloadable application on any computer with a Java 1.1 or later virtual machine. Downloadable distributions are available for Windows, Mac OS, Mac OS X and Linux.

The plugins are available at `http://www.pvv.org/~perchrh/imagej/` as free software under the GPL[1] License. The plugins require Java 1.3 or later.

Adaptive 3D Threshold. Thresholds according to a base value and the values beneath a moving mask of radius N (equation (4.1)).

Susan Smoothing. This is an implementation of 2D and 3D Susan smoothing. A c-version that uses less memory and is faster is also available. Both programs are based on the FSL [2] version of Susan.

3D Discrete Distance Transform. Computes the 3D discrete (values 0 to 255) distance transform, distance is distance to background. Based on the algorithm in Borgefors (1996) and code from Centrum för bildanalys. See the source code for more details.

Isolated Pixel Removal. Removes background pixels (black) completely surrounded by foreground pixels (white). Works on both 2D and 3D images (stack). Treats a stack as a 3D image.

3D Otsu thresholding. For each slice in an image stack, the plugins uses Otsu thresholding to find the optimal threshold. The mean, min, max and median of these values are shown as the result. Based on the ImageJ plugin for Otsu thresholding by Christopher Mei (christopher.mei@sophia.inria.fr) and Maxime Dauphin.

Semi band-threshold marker. Thresholds a volume by setting all values under T1 to 0 and all values above or equal to T2 to 255. Values in-between are set to 1.

Semi band-threshold. Thresholds a volume by setting all values under T1 to 0 and all values above or equal to T2 to 255. Values in-between are left unchanged.

[1] Gnu General Public License, `http://www.gnu.org/copyleft/gpl.html`
[2] Fmrib Software Library, `http://www.fimrib.ox.ac.uk/fsl/`

Small 3D region removal Removes 6-connected 3D regions smaller than a specified size in pixels. Uses code from Runar Holen, with permission.

Chapter 8

Concluding remarks

8.1 Conclusion

A new method for binarization of absorption mode X-ray images of paper was suggested and found to be an improvement over certain other methods. The method also performed well on volumes of wood-fibers in epoxy-vinyl matrix, a type of fiber-reinforced composite material. It remains an open question for now whether this method is better than the procedure of Rolland du Roscoat et al. (2005).

A new method for the segmentation of fibers from the above-mentioned types of images was suggested. The method is a simplification of Holen and Hagen (2004), and improves the wall-segmentation by using a fully three-dimensional approach. In addition, it provides a full segmentation along all axes of orientation, is significantly faster and more automated than previous methods. Rules for rejecting false lumen-candidates to use in automatic fiber lumen detection were found. Rules based on convexity, circularity and eccentricity were found to be efficient in this task.

A practical tool for fiber segmentation and measurements has been developed and released to the scientific community.

A publication is planned. Details can be found in the appendix.

8.2 Suggestions for future work

The tools and methods developed in this thesis can be used by material scientists to conduct studies on fibrous materials. Among other studies, it will be used to calculate the orientation distribution from segmented pore volumes.

We have two suggestions on tasks for computer scientists:

8.2.1 Segmentation of fibers without defined lumens

The segmentation method used in this thesis and in Holen and Hagen (2004) both assume that fibers have well-defined lumens. A different method is needed to segment those fibers that have a severely deformed lumen or no lumen at all. A method for segmenting these kinds of fibers was suggested by Lunden (2002). The method used edge-detection to separate individual fibers. A 50% success rate with a high level of user-interaction was reported. It is still the authors' opinion that edge-detection is necessary to segment these fibers. We therefore suggest examining state of the art methods of edge-detection to find the most suitable method for fiber segmentation. It may be beneficial to combine the segmentation by edges with segmentation by texture of the regions within the edges.

8.2.2 Automatic rule learning

Our software tool can be used to do measurements of shape on binarized volumes. The resulting datasets can be used as input to a rule learning algorithm, like C5.4 (Quinlan 1993), to derive rules for separating lumens and non-lumens. This can result in improved rules and a better understanding of the structural properties of lumens through these rules. Part of the dataset should be used as a verification set to check the validity of the generated rules.

Appendix A

Tests

A.1 S8-HH orientation data

ID	x	y	z
1	0.8709	−0.4915	0.0080
2	−0.9036	−0.4192	0.0880
3	−0.9929	0.1162	0.0243
4	−0.6811	−0.0125	0.7321
5	0.9114	−0.1744	0.3727
6	0.3218	−0.2781	0.9050
7	0.0012	−0.2393	0.9710
8	0.6471	0.7624	0.0057
9	−0.9622	0.0058	0.2724
10	−0.9922	0.0489	0.1148
11	0.0756	0.0167	0.9970
12	−0.9123	0.0863	0.4004
13	−0.9448	−0.3271	0.0190
14	0.5654	−0.8245	0.0238
15	0.9353	0.2315	0.2678
16	−0.0496	−0.0226	0.9985
17	0.7858	−0.1710	0.5943
18	−0.6154	0.0702	0.7851
19	−0.7143	−0.6998	0.0076
20	−0.7227	0.1515	0.6743
21	−0.7694	0.6202	0.1533
22	0.2133	−0.2427	0.9464
23	0.9678	0.2085	0.1407
24	−0.0654	0.1054	0.9923
25	−0.9082	−0.2571	0.3301
26	0.2206	−0.7501	0.6234
27	0.2997	−0.1795	0.9370
28	−0.2860	−0.2014	0.9368
29	0.5209	0.0426	0.8526

30	−0.1251	−0.1709	0.9773
31	0.1646	0.1201	0.9790
32	−0.2898	0.0494	0.9558
33	−0.8947	−0.0160	0.4464
34	−0.1323	−0.1232	0.9835
35	0.4610	−0.1354	0.8770
36	0.7985	0.0904	0.5951
37	−0.6421	−0.0090	0.7665
38	0.2909	−0.2033	0.9349
39	−0.9121	0.0462	0.4073
40	0.8398	−0.1158	0.5304
41	0.9117	0.0190	0.4104
42	0.9097	0.0012	0.4153
43	−0.1476	−0.1666	0.9749
44	−0.1577	0.0220	0.9872
45	−0.7037	0.0196	0.7103
46	−0.0812	−0.4959	0.8646
47	−0.2780	−0.2063	0.9382
48	0.8485	−0.3318	0.4123
49	0.3169	−0.0079	0.9484
50	0.8001	−0.0100	0.5998
51	−0.2993	−0.0949	0.9494
52	0.2107	0.0039	0.9775
53	0.9194	−0.1305	0.3710
54	−0.8661	−0.0869	0.4922
55	0.3485	−0.1829	0.9193
56	0.0522	−0.1061	0.9930
57	−0.8139	−0.1520	0.5608
58	0.3563	−0.4946	0.7927
59	0.7015	−0.3111	0.6412
60	0.4508	−0.0436	0.8916
61	0.1174	−0.1164	0.9862
62	−0.2488	−0.0543	0.9670
63	−0.2753	−0.0729	0.9586
64	−0.4810	−0.1019	0.8708
65	−0.7584	0.0563	0.6493
66	−0.0000	0.0027	1.0000
67	0.7771	0.1400	0.6136
68	−0.1900	−0.0468	0.9807
69	0.0930	0.3105	0.9460
70	−0.1357	−0.0990	0.9858
71	0.3930	0.0234	0.9192
72	0.0746	−0.1098	0.9912
73	0.2156	0.0131	0.9764
74	−0.0300	0.0567	0.9979
75	−0.2160	−0.0034	0.9764
76	0.8523	−0.1572	0.4989
77	0.4546	−0.0770	0.8874
78	−0.3853	0.0158	0.9227
79	−0.1000	−0.0048	0.9950
80	−0.4679	−0.1472	0.8714

81	−0.4263	0.0111	0.9045
82	0.4036	−0.0673	0.9125
83	−0.8195	−0.0076	0.5731
84	0.3462	−0.2750	0.8969
85	0.7123	−0.2274	0.6640
86	0.1672	−0.0156	0.9858
87	−0.0231	0.0347	0.9991
88	0.0482	−0.0482	0.9977
89	0.1793	−0.0962	0.9791
90	0.4779	0.0351	0.8777
91	0.4983	−0.0691	0.8642
92	0.4452	−0.0511	0.8940
93	−0.1036	0.0038	0.9946
94	−0.4187	−0.0350	0.9074
95	−0.4064	0.0286	0.9132
96	0.8556	−0.0949	0.5089
97	0.0342	−0.0058	0.9994
98	0.6407	−0.0047	0.7677
99	−0.3031	0.3332	0.8928
100	−0.2789	−0.0314	0.9598
101	0.3157	−0.0519	0.9474
102	0.1456	−0.1195	0.9821
103	0.3029	−0.0240	0.9527
104	0.5039	−0.0606	0.8616
105	−0.2349	0.0889	0.9679
106	0.5876	−0.0654	0.8065
107	−0.0022	−0.0554	0.9985
108	0.2546	−0.1807	0.9500
109	−0.3222	−0.0687	0.9442
110	−0.6624	0.0647	0.7464
111	0.7095	−0.0839	0.6997
112	0.2024	−0.1734	0.9638
113	0.2669	−0.0388	0.9629
114	0.8337	−0.0614	0.5488
115	−0.2533	−0.0572	0.9657
116	0.0876	0.1163	0.9893
117	0.5018	−0.0421	0.8640
118	−0.5732	0.0563	0.8175
119	−0.5282	0.0241	0.8488
120	0.3779	−0.1256	0.9173
121	0.7596	−0.0566	0.6479
122	0.6178	−0.0494	0.7848
123	0.6004	−0.0849	0.7952
124	0.7785	−0.1159	0.6169
125	0.4996	−0.0041	0.8663
126	−0.0027	−0.0201	0.9998
127	−0.3878	−0.0816	0.9181
128	−0.3498	−0.0669	0.9344
129	0.2933	−0.0684	0.9536
130	−0.6112	0.0578	0.7893
131	0.4608	−0.0024	0.8875

132	−0.4803	−0.0258	0.8767
133	0.8348	−0.0502	0.5482
134	0.2200	−0.0291	0.9751
135	−0.0091	−0.0598	0.9982
136	0.2863	−0.0514	0.9568
137	0.1147	−0.1334	0.9844
138	−0.7126	−0.0185	0.7014
139	0.0460	−0.0258	0.9986
140	−0.4095	0.0127	0.9122
141	0.1091	−0.0805	0.9908
142	−0.0696	−0.0304	0.9971
143	0.0718	−0.0492	0.9962
144	−0.4161	0.0267	0.9089
145	−0.3946	0.0412	0.9179
146	−0.5987	0.0390	0.8000
147	−0.0106	−0.0373	0.9992
148	0.0345	−0.0128	0.9993
149	0.1192	−0.0663	0.9906
150	0.3909	−0.0243	0.9201
151	0.3478	−0.0421	0.9366
152	0.1746	−0.0391	0.9839
153	0.2966	−0.0337	0.9544
154	0.2908	−0.0812	0.9534
155	0.2245	−0.0439	0.9735
156	−0.2552	0.0095	0.9668
157	−0.5613	−0.0060	0.8276
158	0.0912	−0.0636	0.9938
159	0.0642	−0.1109	0.9918
160	−0.5157	0.0497	0.8554
161	−0.7580	0.0586	0.6496
162	−0.3451	0.0028	0.9386
163	0.3791	−0.1005	0.9199
164	−0.1765	−0.0255	0.9840
165	0.4788	−0.0413	0.8770
166	0.3884	−0.0220	0.9212
167	0.0711	−0.0374	0.9968
168	−0.1817	−0.0342	0.9828
169	0.1423	0.0003	0.9898
170	0.3413	−0.1077	0.9338
171	0.5365	−0.0714	0.8409
172	0.6490	−0.1212	0.7511

A.2 S8 orientation data

ID	x	y	z
491	0.5336	−0.0026	0.8457
488	−0.2960	0.2584	0.9196
448	0.1153	−0.0977	0.9885
1117	0.8907	0.2180	0.3988

958	0.8008	0.0120	0.5988
800	0.7058	−0.3310	0.6263
602	−0.5087	0.0149	0.8608
769	0.4929	−0.1889	0.8494
777	0.6464	0.3782	0.6626
1068	0.6325	−0.3159	0.7072
1138	−0.6632	−0.3228	0.6753
494	0.0268	−0.1324	0.9908
519	−0.3314	−0.0007	0.9435
492	0.4676	0.0334	0.8833
1123	0.9531	−0.0481	0.2988
266	−0.4341	0.0284	0.9004
533	0.4198	0.0068	0.9076
400	−0.3314	0.1923	0.9237
332	−0.0994	−0.4900	0.8660
395	0.2092	0.0046	0.9779
341	−0.2942	−0.0940	0.9511
913	−0.7921	0.1257	0.5973
361	−0.4201	−0.3648	0.8310
1137	0.4777	0.0827	0.8746
353	−0.3972	0.0779	0.9144
1130	0.9477	−0.3192	0.0053
1134	0.7765	−0.1975	0.5984
357	−0.7179	−0.4361	0.5426
500	0.0414	−0.1285	0.9908
422	0.3813	−0.1924	0.9042
1038	0.9110	−0.0668	0.4070
574	−0.5640	0.0091	0.8257
1119	−0.7712	0.3654	0.5213
715	0.1395	0.0944	0.9857
366	−0.2117	0.3921	0.8952
1036	0.2898	−0.0977	0.9521
450	0.8168	−0.1593	0.5545
1139	0.9887	0.0985	0.1133
327	0.8865	−0.0715	0.4572
468	−0.2522	0.0075	0.9676
540	0.6711	−0.3069	0.6749
349	0.3368	−0.4297	0.8378
600	0.2389	−0.2164	0.9466
1034	−0.0419	0.0068	0.9991
1081	−0.9727	−0.0112	0.2318
1048	−0.7307	−0.1057	0.6744
447	−0.3033	−0.1750	0.9367
668	0.1856	−0.2786	0.9423
1113	0.6472	−0.4614	0.6068
1126	−0.6690	0.0465	0.7418
822	0.8585	−0.0406	0.5112
333	−0.8474	0.1918	0.4951
1129	0.7088	−0.1892	0.6795
804	−0.5086	−0.1012	0.8550
797	−0.1494	−0.0091	0.9887

849	0.7823	−0.0130	0.6228
859	0.9657	−0.2525	0.0601
350	0.9165	−0.1758	0.3593
359	0.3727	−0.1363	0.9179
329	−0.3835	0.0288	0.9231
810	0.9111	0.2911	0.2920
346	−0.4408	−0.0920	0.8929
1136	−0.9751	−0.2084	0.0758
627	−0.0365	−0.0565	0.9977
841	−0.9266	0.0483	0.3730
1116	0.6042	−0.4217	0.6761
365	0.7183	−0.0955	0.6891
969	−0.9305	0.1608	0.3292
351	0.0934	0.3173	0.9437
665	−0.2025	−0.0040	0.9793
313	0.5152	−0.0110	0.8570
466	0.7978	0.0667	0.5993
1127	0.9649	−0.2054	0.1634
426	−0.2474	−0.0121	0.9688
505	−0.0783	−0.0278	0.9965
622	0.8242	−0.0282	0.5657
828	0.8240	−0.1979	0.5309
526	0.8070	−0.0141	0.5904
1000	−0.5406	0.1950	0.8184
932	−0.8929	−0.0776	0.4435
808	0.9969	0.0334	0.0715
363	−0.9965	0.0832	0.0070
379	−0.2983	−0.0550	0.9529
419	0.4109	0.0998	0.9062
547	0.4244	−0.1375	0.8950
324	−0.8453	0.0981	0.5252
290	0.7790	−0.2160	0.5887
1135	−0.3171	−0.0709	0.9457
717	0.2164	−0.1572	0.9636
337	0.4032	−0.0682	0.9126
367	0.0035	−0.1178	0.9930
802	−0.6938	0.0904	0.7144
280	0.4668	0.0357	0.8836
771	0.9384	−0.0001	0.3457
872	0.7187	−0.2265	0.6574
888	0.8798	−0.1387	0.4546
1128	−0.8344	0.1082	0.5404
459	0.2915	−0.1186	0.9492
405	−0.8972	−0.1464	0.4166
401	0.6763	−0.0873	0.7314
656	−0.3971	0.0189	0.9176
312	−0.0619	−0.1088	0.9921
790	0.9569	0.0805	0.2791
369	−0.2563	0.1555	0.9540
406	−0.1792	−0.0567	0.9822
469	−0.0277	0.0346	0.9990

320	0.2156	−0.0284	0.9761
404	0.0433	−0.1091	0.9931
981	0.9833	0.1516	0.1010
487	0.4450	−0.0522	0.8940
275	−0.1177	−0.0018	0.9930
857	−0.4252	−0.0383	0.9043
558	0.0763	−0.3163	0.9456
1115	0.7235	−0.2138	0.6564
330	−0.2691	0.0119	0.9630
781	−0.9940	0.0661	0.0868
403	−0.3055	0.0489	0.9509
1108	0.3002	−0.0540	0.9523
766	−0.8070	0.0495	0.5885
885	−0.7282	−0.0357	0.6844
567	−0.2652	0.3235	0.9083
345	0.1580	−0.0971	0.9827
354	0.0536	−0.0403	0.9977
440	0.3414	−0.0614	0.9379
342	0.2882	−0.5914	0.7531
1015	0.9940	−0.1041	0.0335
308	−0.8034	−0.0232	0.5950
476	−0.2535	0.0073	0.9673
504	−0.3783	0.0341	0.9251
355	0.5903	−0.0633	0.8047
302	0.1489	−0.1178	0.9818
262	−0.2302	0.0910	0.9689
575	−0.7030	0.0132	0.7111
883	−0.9996	0.0167	0.0208
358	0.5037	−0.0595	0.8618
263	−0.3128	−0.0673	0.9474
282	0.2884	−0.0308	0.9570
465	−0.2789	−0.0418	0.9594
317	−0.2586	0.0578	0.9643
687	0.8906	−0.0967	0.4443
347	−0.0373	−0.1250	0.9915
549	0.2029	−0.1741	0.9636
309	0.2608	−0.1453	0.9544
300	0.2631	−0.0392	0.9640
768	0.8321	−0.0574	0.5516
325	−0.1203	−0.2038	0.9716
288	−0.6673	0.0595	0.7424
305	0.0877	0.1145	0.9895
298	0.5033	−0.0436	0.8630
779	0.8648	−0.0589	0.4986
738	−0.5287	0.0249	0.8484
588	0.5272	−0.0331	0.8491
295	−0.7869	−0.0123	0.6170
1114	0.7787	−0.1158	0.6166
545	0.6020	−0.0753	0.7950
348	−0.0014	−0.0818	0.9966
525	0.2640	−0.0542	0.9630

344	−0.3581	−0.0636	0.9315
276	0.2270	−0.0301	0.9734
470	−0.0922	−0.0081	0.9957
265	−0.3785	0.0609	0.9236
310	0.7568	−0.0527	0.6515
326	0.3763	−0.1231	0.9183
301	−0.4799	−0.0251	0.8769
438	−0.6104	0.0552	0.7902
994	−0.8386	0.0318	0.5439
338	0.6081	−0.0420	0.7927
303	0.1547	−0.0372	0.9873
1102	0.9503	−0.0815	0.3004
933	−0.6015	0.0088	0.7989
868	0.7353	−0.0935	0.6713
323	0.2857	−0.0524	0.9569
654	0.7282	−0.1083	0.6767
992	−0.9559	0.0462	0.2901
335	−0.0119	−0.0577	0.9983
397	0.0751	−0.0541	0.9957
273	0.0511	−0.0230	0.9984
360	−0.0725	−0.0301	0.9969
442	0.5155	−0.0268	0.8565
281	0.5145	−0.0318	0.8569
321	0.2297	−0.1158	0.9664
318	0.4516	0.0059	0.8922
291	0.1131	−0.0800	0.9904
724	−0.4095	0.0109	0.9122
260	0.4095	−0.0187	0.9121
258	0.8345	−0.0637	0.5473
292	0.1092	−0.0653	0.9919
268	−0.4142	0.0263	0.9098
315	−0.2011	0.0727	0.9769
536	0.1801	−0.0396	0.9829
388	0.6043	−0.0063	0.7967
289	0.2947	−0.0339	0.9550
907	−0.3797	−0.0565	0.9234
1122	0.9969	−0.0739	0.0274
297	0.0880	−0.0598	0.9943
375	0.7919	−0.0769	0.6059
749	−0.5087	0.0475	0.8596
269	0.4066	−0.0244	0.9133
272	0.0640	−0.1112	0.9917
286	−0.0179	−0.0379	0.9991
421	0.0343	−0.0130	0.9993
257	−0.3444	0.0026	0.9388
383	−0.0012	−0.0217	0.9998
443	−0.1765	−0.0247	0.9840
384	−0.2502	0.0110	0.9681
264	0.3793	−0.1013	0.9197
436	−0.5620	0.0486	0.8257
671	0.8520	−0.0609	0.5201

364	−0.1802	−0.0478	0.9825
319	0.3925	−0.0238	0.9195
285	−0.6567	0.0137	0.7540
304	0.4671	−0.0279	0.8838
1050	0.2617	−0.0605	0.9633
765	−0.1781	−0.0295	0.9836
1140	−0.8592	0.0515	0.5090
259	0.3412	−0.1077	0.9338
267	0.0743	−0.0351	0.9966
764	0.9799	−0.0765	0.1841
287	0.6488	−0.1210	0.7513
1133	0.5628	−0.0773	0.8230
256	0.1689	−0.0016	0.9856

A.3 STFI7 orientation data

ID	x	y	z
1	0.2872	−0.0886	0.9538
2	−0.9705	0.1169	0.2110
3	0.0270	0.9731	0.2289
4	0.1060	−0.0240	0.9941
5	0.1538	−0.2436	0.9576
6	−0.8539	−0.0919	0.5122
7	0.6334	0.5690	0.5244
8	−0.2070	−0.1124	0.9719
9	−0.1458	−0.1752	0.9737
10	0.5181	−0.2717	0.8110
11	0.3961	−0.0575	0.9164
12	−0.0670	−0.0501	0.9965
13	0.5832	0.0213	0.8120
14	0.4868	−0.0995	0.8678
15	0.9005	−0.0678	0.4295
16	0.8169	−0.1336	0.5611
17	−0.7627	0.2873	0.5795
18	0.1297	−0.0949	0.9870
19	0.4384	−0.1608	0.8843
20	0.8824	0.0075	0.4704
21	−0.0705	−0.5366	0.8409
22	0.1570	−0.3206	0.9341
23	0.7890	0.0014	0.6144
24	0.2978	−0.3500	0.8881
25	0.7611	−0.2330	0.6054
26	0.4773	−0.2545	0.8411
27	−0.1720	0.0641	0.9830
28	0.8780	−0.1736	0.4461
29	0.9508	−0.0846	0.2981
30	0.6418	−0.0295	0.7663
31	0.5485	0.0282	0.8357
32	0.9190	−0.3936	0.0208

33	−0.1772	−0.0233	0.9839
34	0.5842	−0.0665	0.8089
35	0.4867	−0.1212	0.8651
36	−0.9835	0.0450	0.1755
37	0.8337	0.0917	0.5446
38	0.1632	−0.0164	0.9865
39	0.5630	−0.1707	0.8086
40	−0.9300	0.2225	0.2926
41	0.7102	0.1502	0.6878
42	0.9072	−0.1093	0.4064
43	0.0746	0.0716	0.9946
44	0.4274	−0.0062	0.9041
45	0.4290	0.0793	0.8998
46	−0.7594	0.0420	0.6492
47	0.3802	0.1343	0.9151
48	0.4449	−0.0372	0.8948
49	0.0092	−0.0651	0.9978
50	0.6673	−0.1391	0.7317
51	0.7519	0.1918	0.6307
52	0.3082	−0.1702	0.9360
53	0.7399	−0.0603	0.6700
54	0.2881	−0.1010	0.9523
55	0.0992	−0.0170	0.9949
56	−0.0030	0.1863	0.9825
57	−0.7405	0.1384	0.6576
58	0.3962	−0.2046	0.8951
59	0.7615	−0.1578	0.6287
60	−0.0733	−0.0230	0.9970
61	0.3953	−0.1043	0.9126
62	0.8053	−0.1193	0.5808
63	0.4564	−0.0080	0.8897
64	0.8921	−0.0398	0.4501
65	−0.5218	−0.0813	0.8492
66	0.8075	−0.0989	0.5816
67	0.5344	−0.1884	0.8240
68	−0.2254	−0.1187	0.9670
69	0.0518	−0.0094	0.9986
70	0.8026	−0.1433	0.5791
71	−0.2703	−0.0758	0.9598
72	0.4820	0.0086	0.8761
73	0.5308	−0.2791	0.8002
74	−0.8649	0.0109	0.5019
75	0.7443	0.0098	0.6678
76	0.7560	−0.1385	0.6398
77	0.1014	−0.0013	0.9948
78	−0.9576	0.1206	0.2616
79	0.6701	0.0545	0.7403
80	−0.5955	−0.0305	0.8028
81	0.9503	−0.0967	0.2961
82	0.7523	−0.0777	0.6543
83	0.9699	−0.1283	0.2071

84	−0.8958	0.1375	0.4227
85	0.6987	−0.0975	0.7087
86	−0.0692	−0.2154	0.9741
87	0.7267	0.1049	0.6789
88	0.5213	−0.1924	0.8314
89	0.7992	−0.1695	0.5767
90	0.4235	−0.1094	0.8993
91	−0.9148	0.0412	0.4018
92	0.3383	−0.1180	0.9336
93	−0.4613	−0.1120	0.8802
94	−0.7266	0.0189	0.6868
95	0.0557	−0.1093	0.9924
96	0.4736	−0.0587	0.8788
97	−0.8735	−0.1216	0.4713
98	−0.1294	−0.0426	0.9907
99	0.8155	−0.1150	0.5672
100	−0.2354	−0.0152	0.9718
101	0.5373	−0.1036	0.8370
102	−0.1585	−0.0005	0.9874
103	−0.9198	0.1149	0.3752
104	0.6605	−0.1219	0.7409
105	−0.7447	0.1585	0.6483
106	0.3238	0.0872	0.9421
107	0.9636	−0.0720	0.2576
108	0.2349	−0.0908	0.9678
109	0.3771	−0.0256	0.9258
110	0.9845	−0.0337	0.1722
111	0.4483	−0.2622	0.8546
112	0.3385	−0.1343	0.9313
113	0.1930	0.0998	0.9761
114	0.9801	−0.1703	0.1021
115	0.6761	−0.0634	0.7341
116	0.3065	−0.1216	0.9441
117	0.1727	−0.0477	0.9838
118	0.5282	−0.0676	0.8464
119	0.4153	−0.0879	0.9054
120	0.0524	−0.1460	0.9879
121	−0.9970	0.0461	0.0622
122	0.5899	−0.0212	0.8072
123	0.2274	−0.1086	0.9677
124	0.6734	−0.1861	0.7155
125	0.3625	−0.0395	0.9312
126	0.8928	−0.0595	0.4465
127	0.4784	−0.1398	0.8670
128	0.2997	−0.0179	0.9539
129	0.3161	−0.0914	0.9443
130	0.7015	−0.0888	0.7071
131	0.7579	−0.0030	0.6524
132	0.0928	−0.1140	0.9891
133	−0.7356	0.0377	0.6764
134	0.7410	−0.0747	0.6674

135	0.2360	−0.0541	0.9702
136	0.4356	−0.1108	0.8933
137	0.8102	−0.1412	0.5689
138	0.5697	−0.1868	0.8003
139	0.3072	−0.0710	0.9490
140	0.2788	−0.0516	0.9590
141	−0.9945	0.0028	0.1046
142	0.3688	−0.0389	0.9287
143	0.4084	−0.0786	0.9094
144	0.3619	−0.0383	0.9314
145	0.3580	−0.1157	0.9265
146	0.4234	−0.0918	0.9013
147	0.8086	−0.1126	0.5775
148	0.4042	−0.1202	0.9067
149	−0.9954	0.0442	0.0851
150	0.5448	−0.0630	0.8362
151	0.4050	−0.1049	0.9083
152	0.4712	−0.0803	0.8784
153	0.3015	−0.1117	0.9469
154	0.1224	−0.0575	0.9908
155	0.8453	−0.0760	0.5289
156	−0.0003	−0.0634	0.9980
157	0.9392	−0.0823	0.3333
158	−0.7586	−0.0052	0.6515
159	0.5202	−0.1124	0.8466
160	0.8070	−0.1166	0.5790
161	0.8153	0.0092	0.5789
162	−0.9985	0.0437	0.0323
163	0.1656	−0.0347	0.9856
164	0.6366	−0.0591	0.7690
165	0.4230	−0.1233	0.8977
166	0.4111	−0.1012	0.9060
167	0.1214	0.0083	0.9926
168	−0.9941	0.0953	0.0523
169	0.2074	−0.0752	0.9754
170	−0.1691	−0.0601	0.9838
171	0.9979	−0.0628	0.0182
172	−0.0442	−0.0685	0.9967
173	0.2225	−0.0276	0.9745
174	0.2714	−0.1194	0.9550
175	0.2203	−0.0850	0.9717
176	−0.2105	−0.0380	0.9769
177	0.2651	−0.0657	0.9620
178	0.1016	−0.0728	0.9922
179	0.4409	−0.0543	0.8959
180	0.4134	−0.0662	0.9081

A.4 STFI9 orientation data

ID	x	y	z

1	0.0101	0.1541	0.9880
2	−0.6309	0.2630	0.7300
3	0.2796	−0.1127	0.9535
4	0.2962	−0.2353	0.9257
5	0.8497	−0.0749	0.5218
6	0.5431	0.1328	0.8291
7	0.3285	−0.0404	0.9436
8	0.0330	0.3281	0.9441
9	−0.9084	0.2504	0.3348
10	0.4554	0.0811	0.8866
11	0.1592	0.0945	0.9827
12	−0.2400	0.4951	0.8350
13	−0.0031	−0.1628	0.9867
14	0.3811	0.1631	0.9101
15	0.1854	0.0900	0.9785
16	−0.2108	−0.0549	0.9760
17	−0.8517	−0.0614	0.5204
18	0.3406	0.0739	0.9373
19	0.4442	−0.0473	0.8947
20	0.1648	0.1644	0.9725
21	0.5960	−0.2918	0.7481
22	0.0441	0.0254	0.9987
23	0.2974	−0.2703	0.9157
24	−0.0206	−0.1308	0.9912
25	0.4220	−0.1442	0.8950
26	0.0512	0.1490	0.9875
27	−0.9767	−0.1093	0.1849
28	0.3006	0.0155	0.9536
29	0.1002	0.0609	0.9931
30	0.1506	−0.1155	0.9818
31	−0.0829	−0.0147	0.9964
32	0.8787	0.1258	0.4605
33	−0.3793	0.0327	0.9247
34	−0.7976	0.0754	0.5985
35	−0.3499	0.0514	0.9354
36	−0.5750	−0.1021	0.8117
37	−0.0135	0.1326	0.9911
38	0.4451	0.0730	0.8925
39	−0.0429	−0.0262	0.9987
40	−0.9191	0.0201	0.3936
41	0.6820	−0.2034	0.7025
42	−0.0775	−0.1095	0.9910
43	0.0262	−0.1506	0.9883
44	−0.8053	0.2154	0.5523
45	−0.1181	−0.2743	0.9544
46	−0.3152	0.2097	0.9255
47	0.1336	−0.0978	0.9862
48	−0.3051	0.1659	0.9378
49	−0.7746	−0.0343	0.6315
50	0.4906	0.0825	0.8675

51	−0.0136	−0.0401	0.9991
52	−0.0431	−0.0650	0.9970
53	−0.4579	0.3665	0.8100
54	−0.6157	0.0957	0.7821
55	−0.3515	−0.0614	0.9342
56	−0.9860	−0.1527	0.0676
57	0.4592	−0.0945	0.8833
58	−0.1821	−0.1048	0.9777
59	−0.7823	−0.1034	0.6143
60	0.0792	−0.0716	0.9943
61	−0.3491	0.0041	0.9371
62	0.8409	−0.0770	0.5357
63	−0.0233	0.9736	0.2273
64	0.0707	0.2296	0.9707
65	−0.1926	−0.0009	0.9813
66	−0.7961	0.0541	0.6028
67	0.4022	0.1263	0.9068
68	−0.4783	0.0963	0.8729
69	0.8122	0.3374	0.4758
70	−0.1747	−0.0811	0.9813
71	−0.1060	0.0206	0.9942
72	−0.5242	−0.0113	0.8515
73	0.4960	−0.1174	0.8603
74	0.7674	−0.1263	0.6286
75	0.1869	0.0727	0.9797
76	0.2036	0.2192	0.9542
77	0.0990	−0.4484	0.8883
78	0.9719	−0.1425	0.1874
79	−0.2002	−0.0341	0.9792
80	0.1990	−0.0179	0.9798
81	0.6186	0.0319	0.7851
82	−0.2836	−0.6715	0.6846
83	−0.1168	0.1174	0.9862
84	0.7445	−0.3201	0.5859
85	0.1982	0.0140	0.9801
86	0.0316	−0.1097	0.9935
87	−0.0291	0.0191	0.9994
88	0.2625	−0.0409	0.9641
89	−0.2056	−0.0458	0.9776
90	−0.2787	0.1258	0.9521
91	−0.3210	0.0683	0.9446
92	−0.4994	−0.0367	0.8656
93	−0.1736	−0.0695	0.9824
94	−0.4638	−0.0967	0.8807
95	0.0974	0.1981	0.9753
96	0.4426	0.1713	0.8802
97	0.0996	−0.1471	0.9841
98	−0.7675	−0.2752	0.5789
99	−0.9052	0.0042	0.4249
100	−0.2170	0.1298	0.9675
101	0.7062	0.1188	0.6980

102	−0.8696	−0.1124	0.4807
103	0.3727	−0.1366	0.9178
104	0.2601	0.0510	0.9642
105	−0.3557	0.0121	0.9345
106	0.6293	−0.0037	0.7772
107	0.7130	0.0727	0.6973
108	0.2678	0.1314	0.9545
109	0.7482	0.0081	0.6634
110	−0.4532	0.0412	0.8905
111	−0.1543	−0.0010	0.9880
112	0.2974	0.0358	0.9541
113	0.6727	0.1633	0.7217
114	0.9933	−0.1152	0.0023
115	−0.2545	−0.1071	0.9611
116	0.3107	−0.0200	0.9503
117	0.2861	−0.2275	0.9308
118	−0.2170	0.1135	0.9695
119	−0.3172	0.2064	0.9256
120	−0.0049	0.1146	0.9934
121	−0.0798	−0.4982	0.8634
122	0.1808	−0.2015	0.9627
123	0.4735	−0.0366	0.8801
124	−0.2963	−0.0552	0.9535
125	−0.1436	0.0188	0.9895
126	−0.2360	0.0009	0.9717
127	−0.4438	0.0480	0.8948
128	0.1817	0.1168	0.9764
129	−0.1278	−0.0647	0.9897
130	0.0746	−0.1442	0.9867
131	0.9559	0.1056	0.2741
132	0.7181	0.1162	0.6862
133	0.1773	−0.1785	0.9678
134	−0.5314	−0.1047	0.8406
135	0.2366	0.0191	0.9714
136	−0.4800	−0.1320	0.8673
137	−0.5333	−0.7989	0.2780
138	0.2765	0.8399	0.4671
139	0.9080	0.0935	0.4085
140	0.5102	0.1545	0.8461
141	−0.2028	0.0712	0.9766
142	0.4882	0.2105	0.8469
143	0.1465	0.5237	0.8392
144	−0.4128	−0.0313	0.9103
145	−0.3897	−0.0281	0.9205
146	0.2722	0.0938	0.9576
147	−0.5259	−0.1783	0.8316
148	0.7594	0.1389	0.6356
149	0.6762	−0.1282	0.7255
150	0.0720	0.1278	0.9892
151	0.7680	0.1884	0.6122
152	0.3640	0.1864	0.9125

153	−0.0031	0.2108	0.9775
154	0.1480	0.0861	0.9852
155	0.6240	0.0612	0.7791
156	−0.3836	−0.0799	0.9200
157	0.9448	0.1003	0.3121
158	0.5643	0.0846	0.8212
159	0.9565	0.0304	0.2902
160	0.1498	0.4539	0.8784
161	0.0850	0.0816	0.9930
162	0.2367	0.0086	0.9716
163	0.1686	−0.0643	0.9836
164	−0.2677	0.0345	0.9629
165	0.9534	0.2571	0.1580
166	−0.1989	0.0002	0.9800
167	0.0125	0.0025	0.9999
168	−0.7554	−0.0395	0.6541
169	−0.1344	0.0113	0.9909
170	0.1938	0.0662	0.9788
171	0.0902	0.0320	0.9954
172	0.0208	−0.1427	0.9896
173	−0.2698	0.1491	0.9513
174	0.6896	−0.6696	0.2758
175	0.0096	−0.0179	0.9998
176	0.9885	0.0921	0.1202
177	0.2531	0.0981	0.9624
178	−0.3049	0.0175	0.9522
179	−0.2020	−0.0579	0.9777
180	0.1758	−0.0409	0.9836
181	0.3873	−0.0161	0.9218
182	−0.0772	0.0797	0.9938
183	−0.3078	0.0135	0.9514
184	0.4532	−0.0602	0.8894
185	0.1087	−0.1318	0.9853
186	−0.3738	−0.0686	0.9250
187	0.1058	−0.1099	0.9883
188	0.6990	0.1068	0.7071
189	−0.4063	−0.0702	0.9111
190	0.6825	0.0596	0.7285
191	0.3464	−0.0348	0.9374
192	−0.3291	0.0343	0.9437
193	0.2022	0.0354	0.9787
194	−0.1524	0.0954	0.9837
195	0.1937	−0.0690	0.9786
196	−0.1817	−0.0175	0.9832
197	−0.2219	−0.0079	0.9750
198	0.2942	0.0288	0.9553
199	0.1050	0.0027	0.9945
200	−0.1589	−0.0186	0.9871
201	0.2300	−0.0330	0.9726
202	0.2768	0.0277	0.9605
203	−0.4263	−0.0059	0.9046

204	0.1273	0.0366	0.9912
205	0.6557	0.1729	0.7350
206	−0.9966	−0.0456	0.0680
207	−0.3664	−0.0937	0.9257
208	0.0573	−0.0137	0.9983
209	0.4038	0.0233	0.9146
210	0.0100	−0.0693	0.9975
211	0.3683	−0.0215	0.9295
212	−0.0219	0.0094	0.9997
213	−0.1455	0.5622	0.8141
214	−0.0119	0.0286	0.9995
215	−0.2347	0.0038	0.9721
216	−0.8599	−0.4801	0.1732
217	−0.6603	−0.0852	0.7462
218	0.1953	0.0471	0.9796
219	−0.4124	−0.1141	0.9038
220	−0.3470	−0.1451	0.9266
221	0.4944	0.0096	0.8692
222	−0.7720	−0.3433	0.5349
223	0.2119	0.0682	0.9749
224	0.0592	0.0047	0.9982
225	−0.0114	0.0250	0.9996
226	−0.0787	0.0292	0.9965
227	0.9325	0.2670	0.2431
228	0.2872	0.0821	0.9543
229	−0.4274	−0.0417	0.9031
230	−0.5304	−0.1611	0.8323
231	0.2314	0.0522	0.9715
232	−0.3318	0.0714	0.9407
233	−0.1402	−0.0119	0.9900
234	0.8485	−0.2318	0.4758
235	0.2312	0.0734	0.9701
236	0.1316	0.0303	0.9908
237	0.3826	0.1102	0.9173
238	−0.1874	−0.1031	0.9769
239	−0.2427	−0.0735	0.9673
240	0.1112	0.0723	0.9912

Appendix B

Program code and algorithms

B.1 Finding threshold from physical measurements

An interesting way of choosing the threshold value of a paper volume is by its porosity. First, the porosity of the material is measured experimentally. Then, tests are performed to find what porosity values (measured digitally) the different threshold values results in. The threshold value that results in the least difference between the digital and experimental measures of the porosity is the optimal threshold.

The algorithm becomes: Create the cumulative, normalized ($[0,1]$) histogram of the volume, called H. Because of the normalization, we have $V = 1$, and therefore $\Phi = V_p = H(t)$, where t is a threshold value. Search the histogram for the value of t where the minimum of $|\Phi_{\text{measured}} - H(t)|$ occurs. This value of t is the threshold. This can be written mathematically as $T_{\text{opt}} = \min_t |\Phi_{\text{measured}} - H(t)|$

Thanks to Jean-Francis Bloch at EFPG[1] for giving us this idea.

B.2 In-lumen skeleton representation of a 3D lumen

This section gives an algorithm for finding the point in a 2D cross-section of a lumen that is the most fitting part of a skeleton that goes through each of the 2D cross-sections of the lumen. The result is one point for each

[1] École Francaise de Papeterie et des Industries Graphiques, Grenoble, France

cross-section that when put together is a 3D skeleton (or vector-path) that represents that lumen. The 3D distance map, dmap is first created. Then the region is searched to find the maximum and minimum x and y coordinates. The point P_{center} is found by (B.1). The skeleton-point for cross-section n is then found by (B.3), where P_k is point number k in the lumen area and $d(P_i, P_j)$ is the euclidean distance between the two points P_i and P_j. The parameter ω is the wiggle-factor, and is intended to be in the range $(0, 1)$. A low ω means that paths were the skeleton point in cross-section n is a long distance from the skeleton point in cross-sections $n+1$ and $n-1$ are allowed. We found $\omega = 0.25$ to be a good choice.

$$P_{\text{center}} = \left(\frac{x_{\max} + x_{\min}}{2}, \frac{y_{\max} + y_{\min}}{2}\right) \tag{B.1}$$

$$P^0_{\text{skel}} = \max_i \frac{\text{dmap}(P_i)}{d(P_i, P_{\text{center}})} \tag{B.2}$$

$$P^n_{\text{skel}} = \max_i \frac{\text{dmap}(P_i)}{(1-\omega)d(P_i, P_{\text{center}}) + \omega d(P_i, P^n_{\text{skel}} - 1} \tag{B.3}$$

B.3 Fiber segmentation

This section contains implementations of the tracking-methods presented i Chapter 5.

```
/**
 * This is the code executed to track all currently selected
     seeds
 */
public void track() {
  int width = control.getStack().getWidth();
  int height = control.getStack().getHeight();
  int depth = control.getStack().getDepth();

  automerge = (1 == control.getPreferences().getInt("autoMerge
      ", 0));

//    Skeleton structure for each fiber based on 2D centroids
  Point[][] skeleton = new Point[control.MAX_FIBERS][depth];

  Collections.sort(seeds); //sort seeds according to size (
      biggest seeds first)
  Point3D s = null;
  Iterator iter = seeds.iterator();
  int count = 0;
  ArrayList mergeList = new ArrayList();
```

```
keepTracking = true;

while (iter.hasNext() && keepTracking && !control.isStopped
    ()) {
  Point3D s = (Point3D) iter.next(); //get next seed
  seedQueue = new LinkedList();
  count++;
  short[] data = control.getSlice(s.z);
  currentId = (short) (control.getNewFiberId());

  //Init seedQue with initial seed
  seedQueue.addLast(new Seed(s.x, s.y, s.z, 1, null, null,
      0));

  short[] lumenMask = new short[data.length];

  int slice = s.z;

  int filledPixels = ImageFunctions.floodFill2D(s.x, s.y,
      width, height, data, lumenMask, currentId);

  short[] lcopy = (short[]) lumenMask.clone();

  seedStillValid = true;
  int length = 0;

  //Trace forward from seed
  while (slice < depth && seedStillValid && keepTracking) {

    removeCrack(slice, lumenMask);
    removeStuff(slice, lumenMask);

    if (automerge)
      skeleton[currentId][slice] = ImageFunctions.
          getAveragePoint(lumenMask, width, height);

    slice++;
  }

  seedStillValid = true;
  slice = s.z - 1;
  lumenMask = lcopy;

  //Trace backward from seed
  while (slice >= 0 && seedStillValid && keepTracking &&
      control.isStopped()) {

    removeCrack(slice, lumenMask);
    removeStuff(slice, lumenMask);

    if (automerge)
      skeleton[currentId][slice] = ImageFunctions.
          getAveragePoint(lumenMask, width, height);
    slice--;
```

```
      }

      if (automerge)
        checkMerges(skeleton);
      //We must recalculate length, since the total length may
          have grown after
      //merges have been applied!
      length = 0;
      for (int z = 0; z < depth; z++) {
        if (skeleton[currentId][z] != null)
          length++;
      }

      if (removeShortFibers) {
        if (length < minimumFiberSize) {
          //Fiber is rejected reclaim space by marking this seed
              as invalid
          for (int z = 0; z < depth; z++) {
            short[] pixels = control.getSlice(z);
            for (int i = 0; i < pixels.length; i++) {
              if (pixels[i] == currentId)
                pixels[i] = FibForsk.INVALID;
            }
          }
          control.releaseFiberId(currentId);
          skeleton[currentId] = new Point[depth];
        }
      }

    }
    seeds.clear(); //Clear seeds marks
  }


/**
   * The wall labelling is based on 3d dilations, while counting
       the
   * amount of added voxels to each region and stopping the
   * growth if the added number of voxels is less than a quarter
       the result
   * of the first dilation.
   */
  private void traceWalls3d() {

    Stack stack = control.getStack();

    int width = stack.getWidth();
    int height = stack.getHeight();
    int depth = stack.getDepth();

    long first = 0;

    long[] startvals = new long[control.MAX_FIBERS];
    long[] lumenvals = new long[control.MAX_FIBERS];
```

```
  for (int i = 0; i < 15; i++) {
    //      reset all non-negative values
    for (int k = 0; k < lumenvals.length; k++) {
      if (lumenvals[k] > 0)
        lumenvals[k] = 0;
    }
    /**
     * Lumenvals contain a list over the voxel-count for each
         dilation and
     * each lumen. Whenever growth should be stopped, the
         value -1 is
     * set corresponding to the lumen-label
     */
    dilate3d_toWhite(control.getStack(), lumenvals);
    for (int k = 0; k < lumenvals.length; k++) {
      if (lumenvals[k] > 0) {
        if (i == 0)
          startvals[k] = lumenvals[k];
        if (lumenvals[k] < startvals[k] / 4) {
          lumenvals[k] = -1;
        }
      }
    }
  }

  FibForsk control = FibForsk.getInstance();
  for (int z = 0; z < depth; z++) {
    short[] data = (short[]) control.getSlice(z);
    for (int x = 0; x < width; x++) {
      for (int y = 0; y < height; y++) {
        int index = x + y * width;
        if (data[index] == 255)
          continue;
        if ((data[index] & (1 << 12)) != 0) {
          data[index] = (short) (data[index] & ~(1 << 12));
        } else {
          data[index] = 0;
        }
      }
    }
  }
}

/*
 * Returns a factor describing the percentage of border-pixels
     that are adjacent to a white border-area
 * A too low percentage is a strong indication that this
    region is not a valid fiber
 */
public float checkReg(int xp, int yp, final short[] input){
  // implementation details intentionally left out
}
```

```
public boolean removeCrack(int slice, short[] lumen) {
  int width = control.getStack().getWidth();
  int height = control.getStack().getHeight();
  int depth = control.getStack().getDepth();
  boolean isCracked = false;
  short[] data = control.getSlice(slice);
  short[] invdata = ImageFunctions.invertMask(data, false);
  short[] invlumen = ImageFunctions.invertMask(lumen, false);
  Rectangle bounds = ImageFunctions.getBounds(lumen, width,
      height, 12);
  //End search if fibre is outside volume (bounds==null
      indicates no pixels found)
  if (bounds == null || bounds.x < 0 || bounds.y < 0 || bounds
      .width > width || bounds.height > height) {
    seedStillValid = false;
    slice = depth;
    return true;
  }

  if (map == null) {
    map = new short[data.length];
    map2 = new short[data.length];
  }

  //Create a distancemap for use by backtracking, where lumen=
      distance 1
  for (int i = 0; i < map.length; i++) {
    if (lumen[i] != 0)
      map[i] = 1;
    else
      map[i] = 0;
  }
  map = ImageFunctions.dilate(lumen, map, width, height,
      bounds, (short) 2);
  int maxDilations = control.getPreferences().getInt("
      maxDilations", 12);
  for (int i = 3; i < maxDilations; i++) {
    System.arraycopy(map, 0, map2, 0, map.length);
    map = ImageFunctions.dilateMasked(map2, map, invdata,
        width, height, bounds, (short) i);
  }
  map = ImageFunctions.mask(map, invdata, true);

  //Heal cracks detected based on distances with few
      occurances in the map
  BackTrack.backTrack(map, 9, width, height, bounds);

  //copy new lumen back to source
  for (int i = 0; i < map.length; i++) {
    lumen[i] = map[i];
    if (lumen[i] != 0)
      data[i] = currentId;
  }
  return false;
```

```
}

/**
 * Counts the number of lumen regions and orders them
 * by size. Only the largest region is kept, unless the two
 * largest regions are comparable in size indicating a lumen
     split.
 *
 */
public void removeStuff(int slice, short[] lumen) {
  int width = control.getStack().getWidth();
  int height = control.getStack().getHeight();
  int depth = control.getStack().getDepth();
  int regId = 1;
  short[] mark = new short[width * height];
  short[] slicedata = control.getSlice(slice);
  ArrayList liste = new ArrayList();
  for (int x = 0; x < width; x++) {
    for (int y = 0; y < height; y++) {
      if (mark[x + y * width] == 0 && lumen[x + y * width] !=
          0) {
        regId++;
        int size = ImageFunctions.floodFill2D(x, y, width,
            height, lumen, mark, (short) regId);
        LumenCandidate thisCandidate = new LumenCandidate(x, y
            , 0, size);
        float borderRatio = checkReg(x, y, slicedata);
        if (borderRatio < 0.7) {
          //no pixels have a white border, so we remove it!
          ImageFunctions.floodFill2D(x, y, width, height,
              slicedata, (short) 0);
          ImageFunctions.floodFill2D(x, y, width, height,
              lumen, (short) 0);
        } else {
          liste.add(thisCandidate);
        }
      }
    }
  }

  int numberOfRegionsToKeep = 2;

  if (liste.size() > 1) {
    Collections.sort(liste);
    boolean useMR_rules = true;
    if (useMR_rules) {
      LumenCandidate A = (LumenCandidate) liste.get(0);
      LumenCandidate B = (LumenCandidate) liste.get(1);
      //bruk maritrunar-regler
      if (A.getSize() > 5 * B.getSize()) {
        //          System.out.println("Deleted small lumen-
            part.");
        numberOfRegionsToKeep--;
      }
```

```
      }
      for (int i = numberOfRegionsToKeep; i < liste.size(); i++)
          {
        LumenCandidate l = (LumenCandidate) liste.remove(i);
        ImageFunctions.floodFill2D(l.x, l.y, width, height,
            slicedata, (short) 0);
        ImageFunctions.floodFill2D(l.x, l.y, width, height,
            lumen, (short) 0);
      }
    }
  }

  private void checkMerges(Point[][] skel) {
    int width = control.getStack().getWidth();
    int height = control.getStack().getHeight();
    int depth = control.getStack().getDepth();
    int[] mergeList = new int[control.MAX_FIBERS];
    for (int z = 1; z < depth - 1; z++) {
      short[] slice = control.getSlice(z);
      short[] next = control.getSlice(z + 1);
      short[] prev = control.getSlice(z - 1);
      for (int x = 1; x < width - 1; x++) {
        for (int y = 1; y < height - 1; y++) {
          int index = x + y * width;
          if (slice[index] == currentId) {
            mergeList[next[index]] = 1;
            mergeList[prev[index]] = 1;
          }
        }
      }
    }
    for (short color = 1; color < control.MAX_FIBERS; color++) {
      if (isReserved(color))
        continue;
      if (mergeList[color] != 0 && color != currentId) {
        double avgDist = 0;
        double count = 0;
        for (int i = 0; i < depth; i++) {
          if (skel[currentId][i] != null && skel[color][i] !=
              null) {
            double a = skel[currentId][i].x - skel[color][i].x;
            double b = skel[currentId][i].y - skel[color][i].y;
            double dist = Math.sqrt(a * a + b * b);
            avgDist += dist;
            count++;
          }
        }
        if (count > 0)
          avgDist /= (double) count;

        // Only merge regions of they have an average distance
        //across all coinciding lumen sections below this
            threshhold
        if (avgDist < 50 || count == 0) {
```

```
            control.releaseFiberId((short) color);
            for (int z = 0; z < depth; z++) {
              short[] slice = control.getSlice(z);
              for (int x = 1; x < width - 1; x++) {
                for (int y = 1; y < height - 1; y++) {
                  int index = x + y * width;
                  if (slice[index] == color) {
                    slice[index] = currentId;
                  }
                }
              }
              skel[currentId][z] = ImageFunctions.getAveragePoint(
                  slice, width, height, currentId);
              skel[color][z] = null;
            }
          }
        }
      }
    }
  }
}
```

Appendix C

Publications

An article is planned for submission to "Nuclear Instruments and Methods in Physics research", where similar articles have been published previously. The article will deal with segmentation and orientation distribution measurements, and will be co-authored by material scientists at PFI.

An article by Holen and Hagen, whom we based our work on is planned for submission shortly. The title is "Individual segmentation of paper fibers in absorption mode X-ray micro tomographic images". We will wait for them to publish first, as they have worked on this article for quite some time now. Our article will be a direct follow-up to theirs.

We expect their article to be published autumn 2005, and ours to be published early spring 2006.

Bibliography

Antoine, C., Nygaar¿$\frac{1}{2}$d, P., Holmstad, R., Gregersen, O. W., Weitkamp, T., Rau, C., Solheim, O. and Houen, P.-J.: 2001, Binarisation of 3d images of paper obtained by phase-contrast x-ray microtomography, *Research techniques for tomorrow's papermaking*, pp. 1–4.

Arns, C., Pinczweski, W., Knackstedt, M. and Lindquist, W.: 2001, Accurate estimation of transport properties from microtomographic images, *Geophysical Research Letters* **28**(17), 3361–3364.

Aronsson, M.: 2002, *On 3D Fibre Measurements of digitized paper — from microscopy to fibre network*, PhD thesis, Swedish University of Agricultural Sciences.

Bache-Wiig, J. and Henden, P. C.: 2004, Measurements on microtomographic images of fibrous structures. Master-level student project, available at `http://www.pvv.org/~perchrh/papers/`.

Borgefors, G.: 1996, On digital distance transforms in three dimensions, *Comput. Vis. Image Underst.* **64**(3), 368–376.

Borgefors, G. and Svensson, S.: 2001, Optimal local distances for distance transforms in 3d using an extended neighbourhood, *IWVF-4: Proceedings of the 4th International Workshop on Visual Form*, Springer-Verlag, London, UK, pp. 113–122.

Digabel, H. and Lantuejoul, C.: 1978, Iterative algorithms, *Proceedings of 2nd European Symposium Quantitative Analysis of Microstructures in Material Science, Biology, and Medicine, Caen, France, 1977*, Riederer Verlag, Stuttgart, Germany, pp. 85–99.

Fellers, C. and Norman, B.: 1998, *Pappersteknik*, 3. edn, Avdelningen för pappersteknik KTH.

Flynn, P. and Jain, A.: 1989, On reliable curvature estimation, *Proceedings CVPR '89., IEEE Computer Society Conference on*, pp. 110–116.

G. Zak, C. B. P. and Benhabib, B.: 2001, Estimation of three-dimensional fibre-orientation distribution in short-fibre composites by a two-section method, *Journal of Composite Materials* **35**, 316–339.

Garboczi, E., Bentz, D. and Martys, N.: 1999, Digital images and computer modelling, *Experimental Methods for Porous Media*, Academic Press, New York, pp. 1–41.

Gonzalez, R. and Woods, R.: 2001, *Digital image processing*, 2. edn, Prentice Hall.

Holen, R. and Hagen, M.: 2003, Pre-processing and segmentation of micro tomographic images of paper. Master-level student project.

Holen, R. and Hagen, M.: 2004, *Segmentation on absorption mode x-ray micro tomographic images of paper*, Master's thesis, Norwegian University of Science and Technology.

Holmstad, R.: 2004, *Methods for paper structure characterization by means of image analysis*, PhD thesis, Norwegian University of Science and Technology.

Holmstad, R., Antonie, C., Nygaar¿$\frac{1}{2}$d, P. and Helle, T.: 2003, Quantification of the three-dimensional paper structure: Methods and potential, *Pulp & Paper Canada* **104**(7), 186–189.

Huang, S., Goel, A., Ramaswamy, S., Ramarao, B. V. and Choi, D.: 2002, Transverse and in-plane pore structure characterization of paper, *Appita Journal* **55**(3), 230–234.

Jain, A. K. and Dubuisson, M.-P.: 1992, Segmentation of x-ray and c-scan images of fiber reinforced composite materials, *Pattern Recogn.* **25**(3), 257–270.

Kirbas, C. and Quek, F.: 2004, A review of vessel extraction techniques and algorithms, *ACM Comput. Surv.* **36**(2), 81–121.

Kvestad, A. P.: 2002, *Filtering and segmentation of paper tomograms*, Master's thesis, Norwegian University of Science and Technology.

Labiche, J. C., Segura-Purchades, J., Brussel, D. and P.Moy, J.: 1996, Frelon camera: Fastreadlownoise, *ESRF NewsLetter* **25**, 41–43.

Lin, K. C.: 2003, Fast image thresholding by finding the zero(s) of the first derivative of between-class variance, *Mach. Vision Appl.* **13**(5-6), 254–262.

Lunden, J.: 2002, *Image analysis methods for evaluation of fibre dimensions in paper cross-sections*, Master's thesis, The Swedish Royal Institute of Technology. `http://www.nada.kth.se/kurser/kth/exjobb/dokument/Rapporter02/lunden_johanna.pdf`.

Mackworth, S. and Mokhtarian, F.: 1988, The renormalized curvature scale space and the evolution properties of planar curves, *Proceedings CVPR '88., Computer Society Conference on*, pp. 318–326.

Marshall, D.: 1997, On-line compendium of computer vision. `http://homepages.inf.ed.ac.uk/rbf/CVonline/`.

Oh, W. and Lindquist, B.: 1999, Image thresholding by indicator kriging, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **21**, 590–602.

Otsu, N.: 1979, A threshold selection method from gray–level histograms, *IEEE SMC* **9**(1), 62–66.

Quinlan, J.: 1993, C4.5 : Programs for machine learning.

Ramaswamy, S., Huang, S., Goal, A., Cooper, A., Choi, D., Badyopadhyay, A. and Ramarao, B.: 2001, The 3d structure of paper and its relationship to moisture transport in liquid and vapor forms, *The science of paper making*, pp. 1289–1311, 1641–1650. Transactions of the 12th fundamental research symposium. Vol. 2&3.

Rasband, W.: 1997-2005, Imagej. `http://rsb.info.nih.gov/ij/`.

Ridler, T. and Calvard, S.: 1978, Picture thresholding using an interactive selection method, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-8, pp. 630–632.

Rolland du Roscoat, S., Bloch, J. and Thibault, X.: 2005, Synchrotron radiation microtomography applied to investigation of paper, *Journal of Physics D: Applied Physics* **38**(10A), A78–A84. `http://stacks.iop.org/0022-3727/38/A78`.

Russ, J. C.: 1995, *The image processing handbook*, CRC Press, Inc., Boca Raton, FL, USA.

Shih, F. Y. and Wu, Y.-T.: 2004, The efficient algorithms for achieving euclidean distance transformation, *IEEE Transactions on image processing* **13**(8).

Skocir, P., Marusic, B. and Tasic, J.: 2002, A three-dimensional extension of the susan filter for wavelet video coding artifact removal, *11th Mediterranean Electrotechnical Conference*, pp. 395–398.

SkyScan: 2005, Skyscan-2011 nanotomograph. `http://www.skyscan.be`.

Smith, S. and Brady, J.: 1997, SUSAN - a new approach to low level image processing, *Int. Journal of Computer Vision* **23**(1), 45–78.

Sollie, P.: 2003, *Morphological Image Analysis*, 2. edn, Springer.

Sonka, M., Hlavac, V. and Boyle, R.: 1999, *Image Processing, Analysis, and Machine Vision*, 2. edn, Thomson Publishing inc.

team, T. J.: 1999-2005, Jama: A java matrix package. `http://math.nist.gov/javanumerics/jama/`.

Tizon, X. and Smedby, O.: 2002, Segmentation with gray-scale connectedness can separate arteries and veins in mra, *Journal of Magnetic Resonance Imaging* **15**(4), 438–445.

Tubaas, T.: 2005, *Iso-surface extraction and visualisation of binary 3d x-ray images of fiber structures*, Master's thesis, the Norwegian University of Science and Technology.

Vincent, L. and Soille, P.: 1991, Watersheds in digital spaces: An efficient algorithm based on immersion simulations, *IEEE PAMI, 1991* **13**(6), 583–598.

Wang, L., Ramaswamy, S. and Ramarao, B.: 2004, Surface structure characterization of porous materials using x-ray micro computed tomography, *The 2004 Progress in Paper Physics Seminar*, pp. 54–62.

Weisstein, E. W.: 2005, Curvature. From MathWorld–A Wolfram Web Resource, `http://mathworld.wolfram.com/Curvature.html`.

Wikipedia: 2005, Wikipedia, the free encyclopedia. `http://www.wikipedia.org`.

Yang, H.: 2001, *A geometric and Statistical Analysis of Fibrous Materials from Three-Dimensional High Resolution Images*, PhD thesis, State University of New York.

Yang, H. and Lindquist, W.: 2000, Three-dimensional image analysis of fibrous materials, *Applications of Digital Image Processing*, Vol. 13, SPIE, pp. 275–282.

Young, I., Walker, J. and Bowie, J.: 1974, An analysis technique for biological shape. i, *Information and Control* **25**, 357–370.