OPTIMIZED STRICT MULTISCALE FRANGI PREFILTERING FOR SEGMENTATION: TOWARDS AN AUTOMATED PLACENTAL CHORIONIC SURFACE VASCULAR NETWORK EXTRACTION

A THESIS

Presented to the Department of Mathematics and Statistics

California State University, Long Beach

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Applied Mathematics

Committee Members:

Jen-Mei Chang, Ph.D. (Chair)
James von Brecht, Ph.D.
William Ziemer, Ph.D.

College Designee:

Tangan Gao, Ph.D.

By Lucas Allen Wukmer

B.S., 2013, University of California, Los Angeles

May 2019

WE, THE UNDERSIGNED MEMBERS OF THE COMMITTEE,

HAVE APPROVED THIS THESIS

OPTIMIZED STRICT MULTISCALE FRANGI PREFILTERING FOR SEGMENTATION: TOWARDS AN AUTOMATED PLACENTAL CHORIONIC SURFACE VASCULAR NETWORK EXTRACTION

By

Lucas Allen Wukmer

COMMITTEE MEMBERS

Jen-Mei Chang, Ph.D. (Chair)	Mathematics and Statistics
James von Brecht, Ph.D.	Mathematics and Statistics
William Ziemer, Ph.D.	Mathematics and Statistics

ACCEPTED AND APPROVED ON BEHALF OF THE UNIVERSITY

Tangan Gao, Ph.D.
Department Chair, Mathematics and Statistics

California State University, Long Beach

May 2019

ABSTRACT

OPTIMIZED STRICT MULTISCALE FRANGI PREFILTERING FOR SEGMENTATION: TOWARDS AN AUTOMATED PLACENTAL CHORIONIC SURFACE VASCULAR NETWORK EXTRACTION

By

Lucas Allen Wukmer

May 2019

Recent statistical analysis of placental features has suggested the usefulness of studying key features of the placental chorionic surface vascular network (PCSVN) as a measure of overall neonatal health [1]. A recent study has suggested that reliable reporting of these features may be useful in identifying risks of certain neurodevelopmental disorders at birth. The necessary features can be extracted from an accurate tracing of the surface vascular network, but such tracings must still be done manually, with significant user intervention. Automating this procedure would not only allow more data acquisition to study the potential effects of placental development on later conditions, but even perhaps provide a real-time diagnostic for neonatal risk factors.

Much work has been to develop reliable vascular network segmentation methods for well-known image domains (such as retinal MRA images) using Hessian-based filters, namely the multiscale Frangi filter. It is desirable to extend these techniques to study placental images, but the approach has been historically hindered by the comparative irregularity of the placental surface as a whole, which introduces significant noise into filtered result. Previous work has either involved additional filtering [2] or other techniques that are often time-consuming and resource intensive [3].

Here we provide an in-depth mathematical background of the multiscale Frangi filter.

Informed by this theory, we are able to identify stricter parameters that allow us to greatly improve our result. We also reimplement the Frangi filter in frequency space (using a fast fourier transform), which allows us to quickly probe many scales.

We demonstrate the effectiveness of our sped-up implementation of the Frangi filter by performing a large (twenty scales) multiscale Frangi filter on a subset of 201 placental images from a private database provided by the National Children's Study (NCS). We then compare several approaches of merging the multiscale result into an approximation of the PCSVN and compare them to manual tracings of the network. Finally, we develop the notion of the *signed* Frangi filter, upon which we describe a novel-yet-straightforward segmentation method called "trough filling".

ACKNOWLEDGEMENTS

Thank you to the committee for their patience. Thank you to my family and friends for their support. Thank you to you, the reader, for reading this. This work is dedicated to my mother, who always supported me and encouraged me to finish my thesis and get it over with already.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

CHAPTER 1

INTRODUCTION

Recent statistical analysis has suggested a link between placental development and incidence of Autism Spectrum Disorder (ASD) [1]. There is some evidence as in [4] of correlation between this risk and placental health factors. Most ASD cases are not diagnosed until the child reaches three or four, so the benefit of an early detection technique would be massive, as the brain may be more receptive to treatment at a young age. In particular, it was shown in [4] that measurements of the placental chorionic surface vascular network (PCSVN) may be useful in identifying such risk. [1] has provided a method of automatically calculating such features from an extracted vascular network, but does so with manual tracing of the PCSVN in order to make these measurements. These manual tracings are labor-intensive, requiring 4 to 8 hours of user intervention to generate each trace. The present work follows several other efforts towards an automated extraction technique, such as [5], [2], and most recently [3]. Automating this procedure would not only allow more data acquisition to study the potential connection between ASD and placental health, but could even potentially serve as the basis of real-time diagnostic for neonatal risk factors as well. We continue the work of developing a procedure to automate extraction of the PCSVN.

Our basic goal of “vascular network extraction” is a common one in image processing. There are have been many techniques adapted to extracting vascular networks. However, the placenta in particular poses as a particularly difficult image domain to work with. It is a surface network with much irregularity: there are frequent gaps and crossings, and the "background" has a great degree of variation in color intensity and structure itself, causing many naïve approaches at segmentation to fail completely.

Reliable vascular extraction methods do exist for well-known image domains (such as retinal MRA images) using Hessian-based filters, namely the (multiscale) Frangi filter. It is desirable to extend this technique to study placental images, but this approach is greatly hindered by the comparative irregularity of the placental surface as a whole, which introduces significant noise into the image domain. Previous attempts to implement the Frangi filter in this context [2] dealt with this noise by passing the Frangi filtered result to a custom directional filter bank as an additional local curvilinear filter, in an effort to retrieve a partial segmentation. Other more recent efforts are very promising, but are considerably more resource-intensive [3].

Here we provide an in-depth mathematical background of the Frangi filter and its justification as an image-processing technique, as well as an introduction to the Gaussian scale space theory common to many multiscale methods. A more throughout treatment of these underlying theories allows us to choose parameters much more meaningfully than previous efforts. In particular, a more thoughtful selection of the Frangi filter's two parameters reduces much of the noise previously encountered within this image domain. Finally, we discuss an important advancement in implementation–scale space conversion for differentiation (i.e. Gaussian blur) via Fast Fourier Transform, which offers a significant speedup. This allows us faster calculation of the eigenvalues of the Hessian, from which we calculate the Frangi filter, a vesselness measure.

We demonstrate the effectiveness of our sped-up implementation of the Frangi filter by performing a large-scale multiscale Frangi filter on a set of 201 placental images from a private database provided by the National Children's Study (NCS). We then demonstrate the usefulness of this filter by then demonstrating several approaches to segmentation of the PCSVN from our filtered result. These approaches include simple thresholding, scalewise percentile filtering, and finally a custom Frangi-based method we call "trough-filling." We compare each of these segmentation results to the ground truth manual tracings of the network for each of these 201 samples, quantifying the success of various segmentation methods using Matthew's correlation constant (MCC) and precision scoring. Finally, we discuss ways to extend our ideas to solve the

network connection problem.

CHAPTER 2

DIFFERENTIAL GEOMETRY AND SURFACE CURVATURE

Our goal is establish a resource efficient method of finding curvilinear content in 2D grayscale digital images using concepts of differential geometry. We proceed by (i) establishing a standard method of viewing these images as 2D surfaces, (ii) developing a minimal yet rigorous distillation of differential geometry to obtain suitiable quantifiers for the study of curvilinear structure in 3D surfaces, (iii) establishing a filter based on these quantifiers, and finally (iv) developing methods necessary for efficient computation of the filter.

2.1. Problem Setup in Image Processing

A digital 2D grayscale image is given by a $M \times N$ array of pixels, whose intensity is given by an integer value between 0 and 255.

Definition 2.1 (Image as a pixel matrix)**.**

$$\mathtt{I} \in \mathbb{N}^{M \times N} \quad \textit{with} \quad 0 \leq \mathtt{I}_{ij} \leq 2^8 - 1$$

For theoretical purposes, we wish to consider any such picture to ultimately be a sampling of a 2D continuous surface. We also require that this surface is sufficiently continuous as to admit the existence of second partial derivatives.

Definition 2.2 (Image as an interpolated surface)**.**

$$h : \mathbb{R}^2 \to \mathbb{R} \quad \textit{with} \quad h \in C^2\left(\mathbb{R}^2\right), \quad \textit{where} \quad h(i,j) = \mathtt{I}_{ij} \; \forall (i,j) \in \{0,\ldots,M\} \times \{0,\ldots,N\} \subset \mathbb{N}^2$$

That is, the function h is identical to the pixel matrix $\mathtt{I}$ at all integer inputs, and simply a "smooth enough" interpolation of those points for all other values.

It is of course necessary to admit that `I` is not really a perfect representation of the underlying "content" within the picture. Not only is information lost when `I` is stored as an integer, there are also elements of noise and anomalies of lighting that would constitute noise to the original signal. There are multiple treatments of image processing that do address this discrepancy in a pragmatic way [6], especially when the goal is noise reduction. However, we will be content to simply represent the pixels of `I` as the ultimate "cause" of the surface h in Definition 2.2, and worry not about how faithfully that sampling corresponds to the real world. Moreover, though our samples in the image domain have been carefully prepared (as outlined in Section 7.1), there are numerous shortcomings therein, and improvements to the veracity of our original signal could be made from many angles. Though we shall draw upon the notion of the pixel matrix `I` as a sampling again to motivate our development of scale space theory in Chapter 4, we ultimately use these techniques because we find them successful to our problem.

2.2. Differential Geometry

We wish to describe the structure of an image as a surface. To do this, we develop the notion of curvature of a surface in $\mathbb{R}^3$ in a standard way [7].

2.2.1. Preliminaries of Differential Geometry

Given an open subset $U \subset \mathbb{R}^2$ and a twice differentiable function $h : U \to \mathbb{R}$ (as in Definition 2.2) we define the graph, f, of h in the following definition.

Definition 2.3. *The surface f is a graph (of the function h) when*

$$f : U \to \mathbb{R}^3 \quad by \quad f(u_1, u_2) \;=\; \big(u_1, u_2, h(u_1, u_2)\big)\,, \quad u = (u_1, u_2) \in U \subset \mathbb{R}^2$$

Since the graph f is clearly one-to-one (into $\mathbb{R}^3$) by definition, we may readily associate any input $u \in U$ with its corresponding output $p \in f[U]$, i.e.
$p = f(u) = f(u_1, u_2) \;=\; \big(u_1, u_2, h(u_1, u_2)\big)$, depending on whether we wish to focus on a point of a graph in terms of its input or in terms of the structure of the graph itself.

Our development of curvature ultimately will hinge upon a careful consideration of the tangent plane of f at a point p, for we will require a concrete definition of both the tangent space within the domain and image of f, as well as the differential" of f,

Definition 2.4 (Tangent space of U at u)**.**

$$T_u U = \{u\} \times \mathbb{R}^2$$

Definition 2.5 (Tangent space of $\mathbb{R}^3$ at p)**.**

$$T_p \mathbb{R}^3 = \{p\} \times \mathbb{R}^3$$

It is immediately clear that $T_u U$ and $T_p \mathbb{R}^3$ are isomorphic to $\mathbb{R}^2$ and $\mathbb{R}^3$, respectively, and we can easily visualize elements of $T_u U$ are tangent vectors in $\mathbb{R}^2$ "originating" at the point u, and elements of $T_p \mathbb{R}^3$ are tangent vectors "originating" at the point p.

Definition 2.6 (The differential of f at a point u)**.** *$Df|_u$ is the map from $T_u U$ into $\mathbb{R}^3$ given by*

$$Df|_u : T_u U \to T_{f(u)} \mathbb{R}^3 \quad by \quad w \mapsto J_f(u) w$$

where $J_f(u)$ is the Jacobian of f evaluated at some fixed point $u \in U$, i.e. the matrix

$$J_f(u) = \left[\left. \frac{\partial f_i}{\partial u_j} \right|_u \right]_{i,j}$$

We could also just identify the differential Definition 2.6 with a mapping $U \to \mathbb{R}^3$ by the obvious isomorphism inherent in Definition 2.4. In this case, the differential of f at x is simply a linear transformation between the tangent spaces $T_u U$ and $T_p \mathbb{R}^3$ where the matrix of the transformation is given by the Jacobian. We can define such a differential at any point u in the domain.

With these three definitions, we are equipped to give a formal definition of $T_u f$, the tangent plane of f at an input u.

Definition 2.7 (Tangent plane of a graph)**.**

$$T_u f := Df|_u(T_u U) \subset T_{f(u)}\mathbb{R}^3 = T_p\mathbb{R}^3$$

The vectors in this plane can thus be identified as tangent vectors from $T_u U$ that have been passed through the differential mapping $Df|_u$. We shall denote a generic tangent vector $X \in T_u f$ at point p. We may expand any such vector X in terms of the basis $\left\{\frac{\partial f}{\partial u_i}\right\}_{i=1,2}$; that is, $\text{span}\left\{\frac{\partial f}{\partial u_1}, \frac{\partial f}{\partial u_2}\right\} = T_u f$. Each of these basis vectors are the columns of the Jacobian matrix and are *parametric* derivatives of f for the input u.

Given the level of abstraction above, it may be refreshing to explicitly show the linear independence of this set in the case of an arbitrary graph f.

Lemma 2.1. *When f is a graph, for all points $u \in U$, $\left\{\frac{\partial f}{\partial u_1}, \frac{\partial f}{\partial u_2}\right\}$ is in fact a basis for the tangent plane $T_u f$.*

Proof. Given the definition of a graph f as in Definition 2.3, we can directly calculate the parametric derivatives of f at a point u.

$$f_{u_1} = (1, 0, h_{u_1}(u)) \quad \text{and} \quad f_{u_2} = (0, 1, h_{u_2}(u))$$

which are obviously linearly independent. Then $Df|_u(1,0) = f_{u_1}$, and $Df|_u(0,1) = f_{u_2}$, which shows $\left\{\frac{\partial f}{\partial u_1}, \frac{\partial f}{\partial u_2}\right\} \in T_u f$. Thus $\left\{\frac{\partial f}{\partial u_1}, \frac{\partial f}{\partial u_2}\right\}$ is a linearly independent subset of $T_u f$, and can serve as its basis. □

Quite obviously, we're assuming $(1,0), (0,1) \in U$. If this is not the case, we pick some α small enough so that $(\alpha, 0)$ and $(0, \alpha)$ are contained and this scaled version would serve as a basis instead.

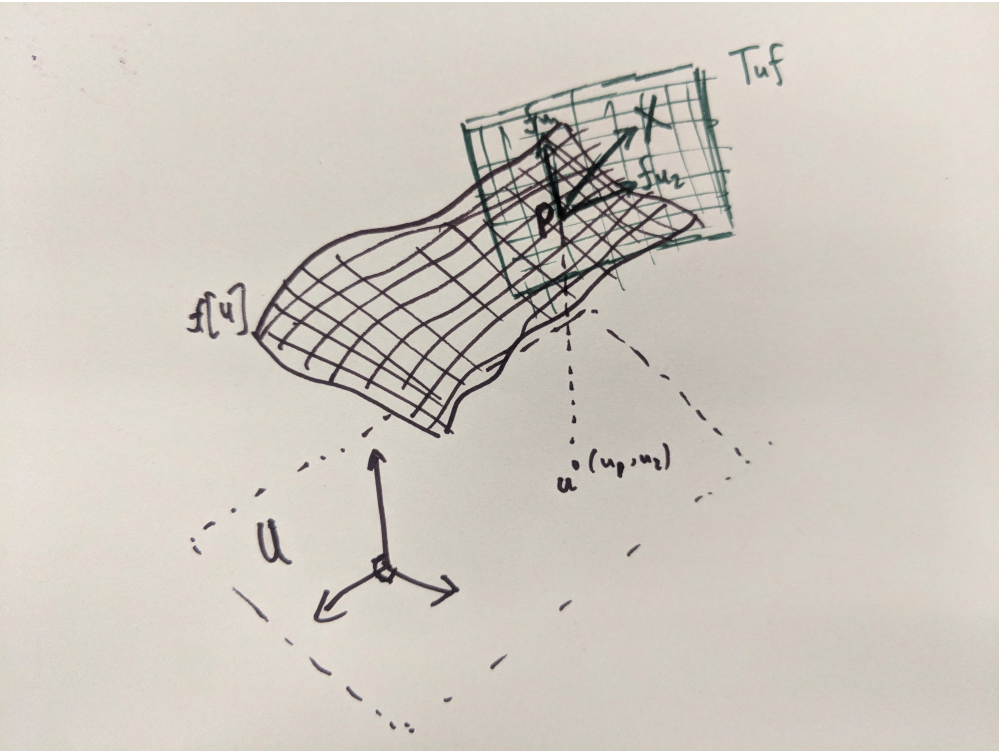


FIGURE 1: Tangent plane of a graph

The parametric derivatives of f are not, in general, orthogonal at any point u, unless it happens that h_{u_1} or h_{u_2} is zero. A visualization of some of the above is given in Fig. 1, although note that f_{u_1} and f_{u_2} accidentally appear orthogonal.

We now concern ourselves with developing the notion of curvature on a surface. First, we need to consider an arbitrary regular curve (i.e. differentiable, one-to-one, non-zero derivative) contained within the image of f.

2.2.2. Curvature of a surface and its calculation

In the context of a regular arc-length parametrized curve $c : I \to \mathbb{R}^3$ parametrized along some closed interval $I \in \mathbb{R}$ (that is, a differentiable, one-to-one curve where $c'(s) = 1 \ \ \forall s \in I$), curvature at a point $s \in I$ is defined simply as the magnitude of the curve's acceleration: $\kappa(s) := \|c''(s)\|$.

To extend the notion of curvature of a surface f, we can consider the curvature of such an

arbitrary curve embedded within the surface.

Definition 2.8 (Surface curve)**.** *Given a closed interval $I \subset \mathbb{R}$, we call the regular curve $c : I \to \mathbb{R}^3$ a surface curve in the event that* $\text{image}(c) \subset \text{image}(f)$ *entirely. The one-to-one-ness of the graph f ensures that we can define (for the given curve) an intermediary parametrization θ so that $c = f \circ \theta$. That is,*

$$\theta : I \to U \text{ by } \theta(t) = \big(\theta_1(t), \theta_2(t)\big)$$

so that $c(t) = f(\theta_c(t)) \; \forall t \in I$, and $c[I] = f\,[\theta_c[I]]$.

Note as well that the velocity of this particular curve lies within $T_u f$. This can be seen by an elementary application of chain rule:

$$\begin{aligned} \frac{dc}{dt} &= \frac{d}{dt}\big[f(\theta_c(t))\big] \\ &= \frac{d}{dt}\big[f(\theta_1(t), \theta_2(t))\big] \\ &= \theta_1'(t)\left(\frac{\partial f}{\partial u_1}\right) + \theta_2'(t)\left(\frac{\partial f}{\partial u_2}\right) \in T_u f. \end{aligned} \tag{2.1}$$

Considering the parameter $p \in I$ and its associated point $u = \theta_c(p)$, we wish to compare the curvatures of all (regular) surface curves passing through the point u at some particular velocity.

We now present a main result that provides a notion of curvature of a surface.

Theorem 2.2 (Theorem of Meusnier)**.** *Given a point $u \in U$ and a tangent direction $X \in T_u f$, any regular curve on the surface $c : I \to \text{image}(f)$ with $p \in I : \theta_c(p) = u$ where $c'(p) = X$ will have the same curvature.*

In other words, any two curves on the surface with a common velocity at a given point on the surface will have the same curvature. To prove this, we require one final definition.

Definition 2.9 (The Gauss Map)**.** *The Gauss map at a point u is the unit normal to the tangent*

plane

$$\nu : U \to \mathbb{R}^3 \quad by \quad \nu(u) := \frac{\frac{\partial f}{\partial u_1} \times \frac{\partial f}{\partial u_2}}{\| \frac{\partial f}{\partial u_1} \times \frac{\partial f}{\partial u_2} \|}$$

Each partial above understood to be evaluated at the input $u \in U$; that is, we calculate $\frac{\partial f}{\partial u_i}\Big|_u$. The existence of the cross product in its definition makes it clear that $\nu \perp \frac{\partial f}{\partial u_i}$ each $i = 1, 2$. A simple dimensionality argument of $\mathbb{R}^3$ implies that these must exist in $T_u f$. However, we can also show it directly: To show that $\left\{\frac{\partial \nu}{\partial u_1}, \frac{\partial \nu}{\partial u_2}\right\} \in T_u f$, first note that at any particular $u \in U$, $\langle \nu, \nu \rangle = 1 \implies \frac{\partial}{\partial u_i}\langle \nu, \nu \rangle = 0$, and so by chain rule $2\langle \frac{\partial \nu}{\partial u_i}, \nu \rangle = 0 \implies \frac{\partial \nu}{\partial u_i} \perp \nu$. Since $\nu \perp \text{span}\left\{\frac{\partial f}{\partial u_i}\right\}$ as well (since ν its outer product), in $\mathbb{R}^3$, this implies $\text{span}\left\{\frac{\partial \nu}{\partial u_i}\right\} = \text{span}\left\{\frac{\partial f}{\partial u_i}\right\}$.

Thus, we have $\text{span}\left\{\frac{\partial \nu}{\partial u_1}, \frac{\partial \nu}{\partial u_2}\right\} \subset T_u f$ as well and we can also use it as a basis. The Gauss map is intrinsic to the surface and irrespective of any particular curve through it.

We are finally ready to prove Theorem 2.2, the Theorem of Meusnier.

Proof. Let $X \in T_u f$ be given and consider some curve where $c'(p) = X$. We wish to decompose the curve's acceleration along the orthogonal vectors X and the Gauss map $\nu = \nu(u_1, u_2) = \frac{\frac{\partial f}{\partial u_1} \times \frac{\partial f}{\partial u_2}}{\| \frac{\partial f}{\partial u_1} \times \frac{\partial f}{\partial u_2} \|}$ as in Definition 2.9. Note that X and ν are indeed orthogonal, as $X \in \text{span}\left\{\frac{\partial f}{\partial u_i}\right\} = T_u f$, and $\nu \perp T_u f$). We then have (at this fixed point $u = \theta(p)$)

$$c'' = \langle c'', X \rangle X + \langle c'', \nu \rangle \nu \tag{2.2}$$

.

Because c is a regular curve, we either have $c'' = 0$, or $c' \perp c''$, since $\|c'\| = 1$ implies $0 = \frac{d}{dt}\langle c', c' \rangle = 2\langle c'', c' \rangle$. Thus

$$\langle c'', X \rangle = \langle c'', c' \rangle = 0$$

and we can rewrite the second coefficient of Eq. (2.2) using the chain rule:

$$\begin{aligned}\langle c'', \nu\rangle &= \frac{\partial}{\partial t}[\langle c', \nu\rangle] - \langle c', \frac{\partial \nu}{\partial t}\rangle \\ &= \frac{\partial}{\partial t}[\langle X, \nu\rangle] - \langle c', \frac{\partial \nu}{\partial t}\rangle \\ &= 0 - \langle X, \frac{\partial \nu}{\partial t}\rangle\end{aligned} \tag{2.3}$$

Thus, we can express the curvature at this point on our selected curve as

$$\begin{aligned}\|c''\| = \|\langle c'', X\rangle X + \langle c'', \nu\rangle \nu\| &= \|0 + \langle c'', \nu\rangle \nu\| \\ &= |-\langle X, \frac{\partial \nu}{\partial t}\rangle| \|\nu\| \\ &= |-\langle X, \frac{\partial \nu}{\partial t}\rangle| \\ &= |\langle X, -\frac{\partial \nu}{\partial t}\rangle|\end{aligned} \tag{2.4}$$

We may compute the quantity $-\frac{\partial \nu}{\partial t}$ that appears in Eq. (2.4) via chain rule:

$$\begin{aligned}-\frac{d\nu}{dt} &= -\frac{d}{dt}\left[\nu(u_1, u_2)\right] \\ &= -\frac{d}{dt}\left[\nu(\theta_1(t), \theta_2(t))\right] \\ &= \theta_1'(t)\left(-\frac{\partial \nu}{\partial u_1}\right) + \theta_2'(t)\left(-\frac{\partial \nu}{\partial u_2}\right)\end{aligned} \tag{2.5}$$

Since span $\left\{-\frac{\partial \nu}{\partial u_i}\right\}_{i=1,2}$ as a subset of $T_u f$, we can identify a linear transformation which maps the basis $\left\{\frac{\partial f}{\partial u_i}\right\}_{i=1,2}$ to this basis.

We call this map which the Weingarten map L.

Definition 2.10 (The Weingarten Map)**.**

$$\mathsf{L} : T_u f \to T_u f \quad \textit{given by the composition} \quad \mathsf{L} = D\nu \circ (Df)^{-1}.$$

That is, $\mathsf{L}(\frac{\partial f}{\partial u_i}) = -\frac{\partial \nu}{\partial u_i}$ for $i = 1, 2$. This allows us to rewrite the time derivative of the Gauss map Eq. (2.5) as

$$\begin{aligned} -\frac{d\nu}{dt} &= \theta_1'(t)\left(-\frac{\partial \nu}{\partial u_1}\right) + \theta_2'(t)\left(-\frac{\partial \nu}{\partial u_2}\right) \\ &= \theta_1'(t)\left(\mathsf{L}\left(\frac{\partial f}{\partial u_1}\right)\right) + \theta_2'(t)\left(\mathsf{L}\left(\frac{\partial f}{\partial u_2}\right)\right) \\ &= \mathsf{L}\left[\theta_1'(t)\left(\frac{\partial f}{\partial u_1}\right) + \theta_2'(t)\left(\frac{\partial f}{\partial u_2}\right)\right] \\ &= \mathsf{L}\left(\frac{d}{dt}\left[f\left(\theta(t)\right)\right]\right) = \mathsf{L}\left(\frac{d}{dt}\left[c(t)\right]\right) = \mathsf{L}(X) \end{aligned} \tag{2.6}$$

With this, we can re-express the curvature of our curve from Eq. (2.4) as the much simplified

$$\|c''\| = |\langle X, -\frac{\partial \nu}{\partial t}\rangle| = |\langle X, \mathsf{L}(X)\rangle| \tag{2.7}$$

The linear transformation L from Definition 2.10, and thereby the computation of curvature given in Eq. (2.7), depends only on the point u and the selected direction X, not on the particular curve c at all. □

To recap, given a point u on the surface and an arbitrary vector X in the tangent plane, we can calculate the curvature of any surface curve with velocity X there. In fact, we refer to this intrinsic quantity as the normal curvature of the surface.

Definition 2.11. *The normal curvature of a surface, denoted κ_ν at point u in the direction X is given by*

$$\kappa_\nu := \langle X, \mathsf{L}(X)\rangle$$

In fact, Theorem 2.2 shows that the normal curvature is an intrinsic property of the surface–it depends only on the surface at a point, and no reference to any particular curve on the surface is necessary or implied.

The map L introduced in the proof above is known as the Weingarten map and is implicitly

defined at each $u \in U$. We wish to make its existence rigorous as well as find a matrix representation for it, using the standard motivation that $\mathsf{L}(\frac{\partial f}{\partial u_i}) = -\frac{\partial \nu}{\partial u_i}$.

That is, we may trace any $X \in T_u f$ which has been expanded in terms of the basis $\left\{\frac{\partial f}{\partial u_1}, \frac{\partial f}{\partial u_2}\right\}$ and map it to the basis $\left\{-\frac{\partial \nu}{\partial u_1}, -\frac{\partial \nu}{\partial u_2}\right\}$.

The Weingarten map can be formally shown to be well-defined, invariant under coordinate transformation in the general case (that is, for surfaces f that are not graphs). We refer to [7] for the general proof. Our present situation is much less delicate, as we're only concerned for cases when f is a graph. In this case, the linear transformation may be simply constructed, and we proceed by simply calculating its matrix representation.

Lemma 2.3. *The Weingarten map as in Definition 2.10 is well-defined for graphs.*

To find a matrix representation for L (which we will denote $\widehat{\mathsf{L}} \in \mathbb{R}^{2\times 2}$), we simply wish to find a linear transformation such that $\widehat{\mathsf{L}}\, \frac{\partial f}{\partial u_i}\Big|_{T_u f} = -\frac{\partial \nu}{\partial u_i}\Big|_{T_u f}$ for $i = 1, 2$ where $-X|_{T_u f}$ denotes that $X \in T_u f$ is being represented in local coordinates for $T_u f$ (Strictly speaking, of course $T_u f \subset \mathbb{R}^3$ and thus $\frac{\partial f}{\partial u_i} \in \mathbb{R}^3$. Thus when we say $\frac{\partial f}{\partial u_i}\Big|_{T_u f}$ we are referring to this 3-vector expanded with respect to the two-dimensional basis for $T_u f$). In matrix form, we describe this situation as

$$\begin{bmatrix} \\ \widehat{\mathsf{L}} \\ \\ \end{bmatrix} \begin{bmatrix} | & | \\ \frac{\partial f}{\partial u_1}\Big|_{T_u f} & \frac{\partial f}{\partial u_2}\Big|_{T_u f} \\ | & | \end{bmatrix} = \begin{bmatrix} | & | \\ \widehat{\mathsf{L}}\frac{\partial f}{\partial u_1}\Big|_{T_u f} & \widehat{\mathsf{L}}\frac{\partial f}{\partial u_2}\Big|_{T_u f} \\ | & | \end{bmatrix} \tag{2.8}$$

$$= \begin{bmatrix} | & | \\ -\frac{\partial \nu}{\partial u_1}\Big|_{T_u f} & -\frac{\partial \nu}{\partial u_2}\Big|_{T_u f} \\ | & | \end{bmatrix} \tag{2.9}$$

Now, representing each vector in $T_u f$ with respect to the basis $\left\{\frac{\partial f}{\partial u_i}\right\}$, we have

$$\begin{bmatrix} \widehat{\mathsf{L}} \end{bmatrix} \begin{bmatrix} | & | \\ \frac{\partial f}{\partial u_1}\Big|_{T_u f} & \frac{\partial f}{\partial u_2}\Big|_{T_u f} \\ | & | \end{bmatrix} = \begin{bmatrix} \widehat{\mathsf{L}} \end{bmatrix} \begin{bmatrix} -\frac{\partial f}{\partial u_1}- \\ -\frac{\partial f}{\partial u_2}- \end{bmatrix} \begin{bmatrix} | & | \\ \frac{\partial f}{\partial u_1} & \frac{\partial f}{\partial u_2} \\ | & | \end{bmatrix} = \begin{bmatrix} -\frac{\partial f}{\partial u_1}- \\ -\frac{\partial f}{\partial u_2}- \end{bmatrix} \begin{bmatrix} | & | \\ -\frac{\partial \nu}{\partial u_1} & -\frac{\partial \nu}{\partial u_2} \\ | & | \end{bmatrix} \tag{2.10}$$

$$\tag{2.11}$$

We can simplify this greatly by defining

$$g_{ij} := \langle \frac{\partial f}{\partial u_i}, \frac{\partial f}{\partial u_j} \rangle \quad \text{and} \quad h_{ij} := \langle \frac{\partial f}{\partial u_i}, -\frac{\partial \nu}{\partial u_j} \rangle \tag{2.12}$$

so that

$$\begin{bmatrix} \widehat{\mathsf{L}} \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} \tag{2.13}$$

Then we rearrange to solve for $\widehat{\mathsf{L}}$ as

$$\widehat{\mathsf{L}} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}^{-1} \tag{2.14}$$

where $\left[g_{ij}\right]$ is clearly invertible, as the set $\left\{\frac{\partial f}{\partial u_j}\right\}$ is linearly independent.

It should be noted that this matrix representation is accurate not only for the surface of a graph, but for any *generalized* surface $f : U \to \mathbb{R}^3$ with $u \mapsto (x(u), y(u), z(u))$ as well. We shall later show that this calculation simplifies (somewhat) in the case that our surface is a graph.

Our final goal is to characterize such normal curvatures. Namely, we wish to establish a method of determining in which directions an extremal normal curvature occurs.

2.2.3. Principal Curvatures and Principal Directions

To do so, we shall consider the relationship between the direction X and the normal curvature κ_ν Definition 2.11 in that direction at some specified u.

First, we need the following lemma:

Lemma 2.4. *If $A \in R^{n\times n}$ is a symmetric real matrix, $v \in R^n$ and given the dot product $\langle\cdot,\cdot\rangle$, we have $\nabla_v\langle v, Av\rangle = 2Av$. In particular, when $A = I$ the identity matrix, we have $\nabla_v\langle v, v\rangle = 2v$.*

Proof. The result is uninterestingly obtained by tracking each component of $\nabla_v\langle v, Av\rangle$:

$$\Big(\nabla_v\langle v, Av\rangle\Big)_i = \frac{\partial}{\partial v_i}\Big[\langle v, Av\rangle\Big] = \frac{\partial}{\partial v_i}\left[\sum_{j=1}^{n} v_j(Av)_j\right] \tag{2.15}$$

$$= \frac{\partial}{\partial v_i}\left[\sum_{j=1}^{n} v_j \sum_{k=1}^{n} a_{jk}v_k\right] \tag{2.16}$$

$$= \frac{\partial}{\partial v_i}\left[a_{ii}v_i^2 + v_i\sum_{k\neq i} a_{ik}v_k + v_i\sum_{j\neq i} a_{ji}v_j + \sum_{j\neq i}\sum_{k\neq i} v_j a_{jk}v_k\right] \tag{2.17}$$

$$= 2a_{ii}v_i + \sum_{k\neq i} a_{ik}v_k + \sum_{j\neq i} a_{ji}v_j + 0 \tag{2.18}$$

$$= 2a_{ii}v_i + 2\sum_{k\neq i} a_{ik}v_k = 2\sum_{k=1}^{n} a_{ik}v_k = 2\big(Av\big)_i \tag{2.19}$$

$$\implies \nabla_v\langle v, Av\rangle = 2Av. \tag{2.20}$$

□

We are now ready for the major result of this section, which ties the Weingarten map to the notion of normal curvatures.

Theorem 2.5 (Theorem of Olinde Rodrigues)**.** *Fixing a point $u \in U$, a direction $X \in T_u f$ minimizes the normal curvature $\kappa_\nu = \langle \mathsf{L}X, X\rangle$ subject to $\langle X, X\rangle = 1$ iff X is a (normalized) eigenvector of the Weingarten map L.*

Proof. In the following, we will assume that $X \in T_u f$ is expanded, in local coordinates, i.e. along a two dimensional basis (such as $\left\{\frac{\partial f}{\partial u_i}\right\}_{i=1,2}$) and thus can refer to L freely as the 2×2 matrix $\widehat{\mathsf{L}}$. Using the method of Lagrange multipliers, we define the Lagrangian:

$$\mathscr{L}(X;\lambda) := \langle \widehat{\mathsf{L}}X, X\rangle - \lambda\Big(\langle X, X\rangle - 1\Big) \tag{2.21}$$

Extremal values occur when $\nabla_{X,\lambda}\mathscr{L}(X;\lambda) = 0$, which results in the two equations

$$\begin{cases} \nabla_X\langle \widehat{\mathsf{L}}X, X\rangle - \lambda\nabla_X\left(\langle X, X\rangle - 1\right) = 0 \\ \langle X, X\rangle - 1 = 0 \end{cases} \tag{2.22}$$

The second requirement is simply the constraint that X is normalized. Using the previous lemma, we can simplify the first result as follows:

$$\begin{aligned} \nabla_X\langle \widehat{\mathsf{L}}X, X\rangle - \lambda\nabla_X\left(\langle X, X\rangle - 1\right) &= 0 \\ 2\widehat{\mathsf{L}}X - \lambda(2X) &= 0 \\ \implies \quad \widehat{\mathsf{L}}X - \lambda X &= 0 \\ \implies \quad \widehat{\mathsf{L}}X &= \lambda X \end{aligned} \tag{2.23}$$

which implies that X is an eigenvector of $\widehat{\mathsf{L}}$ with corresponding eigenvalue λ ($X \neq 0$ from the second equation of Eq. (2.22)). Thus the two hypotheses are exactly equivalent when X is normalized. It is also worth remarking that the corresponding eigenvalue λ is the Lagrangian multiplier itself. □

Thus, to find the directions of greatest and least curvature of a surface at a point $u \in U$, we simply must calculate the Weingarten map and its eigenvectors. We refer to these directions as follows.

Definition 2.12 (Principal Curvatures and Principal Directions)**.** *The extremal values of normal curvature of a surface at a point $u \in U$ are referred to as* ***principal curvatures****. The corresponding directions at which normal curvature attains an extremal value are referred to as* ***principal directions****.*

Our final goal is to explicitly determine a (hopefully simplified) version of the Weingarten map in the case of a graph $f(u_1, u_2) = (u_1, u_2, h(u_1, u_2))$ and calculate the principal directions and curvatures in a simple example.

Theorem 2.6 (Relationship between Hessian and Weingarten Map of a Graph)**.** *Given the graph $f : U \to \mathbb{R}^3$ where $(x, y) \mapsto (x, y, h(x, y))$, the matrix representation of its Weingarten map is given by*

$$\widehat{\mathsf{L}} = \mathrm{Hess}(h)\tilde{G}\,, \quad \text{where} \quad \tilde{G} := \frac{1}{\left(1 + h_x^2 + h_y^2\right)^{3/2}} \begin{bmatrix} 1 + h_y^2 & -h_x h_y \\ -h_x h_y & 1 + h_x^2 \end{bmatrix} \tag{2.24}$$

In particular, given a point $u = (x, y) \in U \subset \mathbb{R}^2$ where $h_x \approx h_y \approx 0$, we have $\tilde{G} \approx \mathrm{Id}$, and thus $\widehat{\mathsf{L}} \approx \mathrm{Hess}(h)$.

Proof. We begin from Eq. (2.14). First, consider each component from Eq. (2.12) and rewrite via chain rule:

$$h_{ij} = \langle \frac{\partial f}{\partial u_i}, -\frac{\partial \nu}{\partial u_j} \rangle = \langle \frac{\partial^2 f}{\partial u_i \partial u_j}, \nu \rangle$$

Now, given our particular surface f, we can calculate each of these components directly. We have:

$$\begin{gathered} f_x = (1, 0, h_x), \quad f_y = (0, 1, h_y) \\ f_{xx} = (0, 0, h_{xx}), \quad f_{xy} = (0, 0, h_{xy}) = f_{yx}, \quad f_{yy} = (0, 0, h_{yy}) \end{gathered} \tag{2.25}$$

and we have the unit normal vector (Gauss map)

$$\nu(u_1, u_2) = \frac{\frac{\partial f}{\partial x} \times \frac{\partial f}{\partial y}}{\|\frac{\partial f}{\partial x} \times \frac{\partial f}{\partial y}\|} \tag{2.26}$$

$$= \frac{(1, 0, h_x) \times (0, 1, h_y)}{\|(1, 0, h_x) \times (0, 1, h_y)\|} \tag{2.27}$$

$$= \frac{\left(-h_x, -h_y, 1\right)}{\sqrt{h_x^2 + h_y^2 + 1}} \tag{2.28}$$

We then calculate each h_{ij} as

$$\begin{aligned} h_{11} &= \langle \frac{\partial^2 f}{\partial x^2}, \nu \rangle = \frac{h_{xx}}{\sqrt{1 + h_x^2 + h_y^2}} \\ h_{12} &= \langle \frac{\partial^2 f}{\partial x \partial y}, \nu \rangle = \frac{h_{xy}}{\sqrt{1 + h_x^2 + h_y^2}} = h_{21} \\ h_{22} &= \langle \frac{\partial^2 f}{\partial y^2}, \nu \rangle = \frac{h_{yy}}{\sqrt{1 + h_x^2 + h_y^2}} \end{aligned} \tag{2.29}$$

and thus the first matrix in Eq. (2.14) is given by

$$[h_{ij}] = \frac{1}{\sqrt{1 + h_x^2 + h_y^2}} \mathrm{Hess}(h) \tag{2.30}$$

To calcuate the second, we use

$$\begin{aligned} g_{ij} &= \langle \frac{\partial f}{\partial u_i}, \frac{\partial f}{\partial u_j} \rangle \\ g_{11} &= \langle f_x, f_x \rangle = 1 + h_x^2 \\ g_{12} &= \langle f_x, f_y \rangle = h_x h_y = g_{21} \\ g_{22} &= \langle f_y, f_y \rangle = 1 + h_y^2 \end{aligned} \tag{2.31}$$

and thus

$$[g_{ij}]^{-1} = \begin{bmatrix} 1+h_x^2 & h_x h_y \\ h_x h_y & 1+h_y^2 \end{bmatrix}^{-1} = \frac{1}{1+h_x^2+h_y^2} \begin{bmatrix} 1+h_y^2 & -h_x h_y \\ -h_x h_y & 1+h_x^2 \end{bmatrix} \tag{2.32}$$

Combining $[h_{ij}]$ and $[g_{ij}]^{-1}$ from Eq. (2.32) and Eq. (2.30) we arrive at our result, Eq. (2.24). □

We stress that this map L is defined for each point $u \in U$. In the particular case that $u \in U$ is a critical point, where $\nabla h(u) = (h_x(u), h_y(u)) = 0$, then it is clear from the previous theorem that $\widehat{\mathsf{L}}$ is exactly the Hessian matrix $\mathrm{Hess}(h)$. Of course this implies that $\widehat{\mathsf{L}}$ and $\mathrm{Hess}(h)$ have the same eigenvalues and eigenvectors at any such point.

But this observation is more broadly useful than for analyzing critical points alone. If $\tilde{G}$ above is close to identity, then the eigenvalues and eigenvectors of $\widehat{\mathsf{L}}$ will be similarly close to the eigenvalues of the Hessian. We can rewrite $\tilde{G}$ from Eq. (2.24) as identity plus a small matrix:

$$\tilde{G} = \frac{1}{1+h_x^2+h_y^2}(I + [\delta]), \quad [\delta] := \begin{bmatrix} h_y^2 & -h_x h_y \\ -h_x h_y & h_x^2 \end{bmatrix} \tag{2.33}$$

We can then rewrite Eq. (2.24) as

$$\widehat{\mathsf{L}} = \frac{1}{\left(1+h_x^2+h_y^2\right)^{3/2}} (\mathrm{Hess}(h) + \mathrm{Hess}(h)[\delta]) \tag{2.34}$$

We can see that as h_x, h_y are close to zero, $[\delta]$ will be very close to the zero matrix, and the constant $\left(1+h_x^2+h_y^2\right)^{-3/2}$ will be very close to 1 as well, so we should not expect the addition of a "close to 0" matrix to have much effect on the eigenvectors or eigenvalues. This intuition is confirmed by a result from Wilkinson [8], which we state without rigorous proof.

Theorem 2.7. *If A, B are matrices such that* $|A_{ij}| < 1, |B_{ij}| < 1$ *(a condition that can be ignored*

with scaling) and λ is a simple eigenvalue of A, then given $\epsilon > 0$, there exists a simple eigenvalue $\tilde{\lambda}$ of the matrix $A + \epsilon B$ with $|\lambda - \tilde{\lambda}| = O(\epsilon)$. Similarly, if v is an eigenvector of A, then $\tilde{v}$ is an eigenvector of $A + \epsilon B$ with $|v - \tilde{v}| = O(\epsilon)$.

The proof ultimately relies on a general result of analysis, that the zeros of a polynomial are continuous with respect to its coefficients. In this case, the polynomial in question is the characteristic polynomial $p(\lambda) = \det(\lambda I - A - \epsilon B)$, whose coefficients will scale with ϵ. Thus $\widehat{\mathsf{L}} \approx \text{Hess}(h)$ for any point where the gradient $\nabla h \approx 0$. We shall see that we're only concerned with regions where h_x, h_y is small anyway, and we do not expect much response anyway when the gradient is large.

We can bound the perturbation of eigenvalues from $\widehat{\mathsf{L}}$ to $\text{Hess}(h)$ by another result which we state without proof [9].

Theorem 2.8. *Let A be a $n \times n$ normal matrix with eigenvalues $\lambda_1, \ldots, \lambda_n$ and E an $n \times n$ arbitrary matrix. If $\hat{\lambda}$ in an eigenvalue of $A + E$, then there is an eigenvalue λ_i of A for which $|\hat{\lambda} - \lambda_i| \leq \|E\|_2$*

In the event that we do wish to rigorously compute the Weingarten map–that is, without concern for the magnitude of the gradient–we refer to [10] and survey papers mentioned therein.

To make the Weingarten map and its relationship to the Hessian more explicit, we will calculate the Weingarten map for a relatively simple graph.

2.2.4. The Weingarten map and Principal Curvatures of a Cylindrical Ridge

Let f be the graph given by

$$f : \mathbb{R}^2 \to \mathbb{R}^3 \text{ by } f(x, y) = (x, y, h(x, y)), \text{ with } h(x, y) = \begin{cases} \sqrt{r^2 - x^2} & -r \leq x \leq r \\ 0 & \text{else} \end{cases} \tag{2.35}$$

The graph is shown in Fig. 2. We calculate the necessary partial derivatives of f as follows:

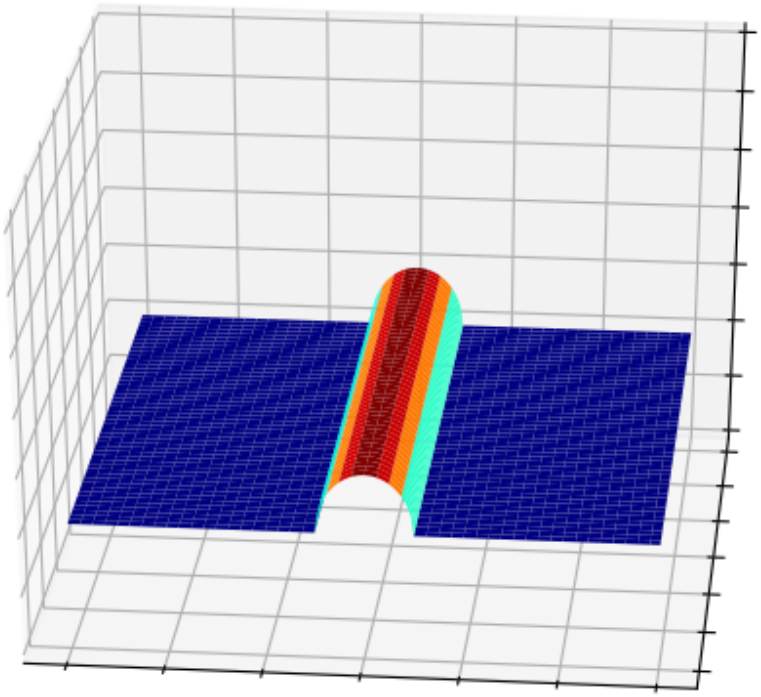


FIGURE 2: The graph of a cylindrical ridge of radius r

$$\frac{\partial f}{\partial x} = \left(1, 0, \frac{-x}{\sqrt{r^2 - x^2}}\right) \quad , \quad \frac{\partial^2 f}{\partial x^2} = \left(0, 0, \frac{-r^2}{\left(\sqrt{r^2 - x^2}\right)^3}\right) \tag{2.36}$$

$$\frac{\partial f}{\partial y} = (0, 1, 0) \quad , \quad \frac{\partial^2 f}{\partial y^2} = \frac{\partial^2 f}{\partial x \partial y} = 0 \tag{2.37}$$

The gauss map is given by

$$\nu(x,y) = \frac{\frac{\partial f}{\partial x} \times \frac{\partial f}{\partial y}}{\| \frac{\partial f}{\partial x} \times \frac{\partial f}{\partial y} \|} = \left(\frac{x}{r},\, 0,\, \frac{\sqrt{r^2 - x^2}}{r} \right) \tag{2.38}$$

$$\implies \frac{\partial \nu}{\partial x} = \left(\frac{1}{r}\,,\, 0\,,\, \frac{-x}{r\sqrt{r^2 - x^2}} \right) \quad , \quad \frac{\partial \nu}{\partial y} = (0,0,0)\,. \tag{2.39}$$

We then calculate matrix elements of the Weingarten map's construction as given in Eq. (2.30) and Eq. (2.32) :

$$[h_{ij}] = \frac{1}{\sqrt{1 + h_x^2 + h_y^2}} \mathrm{Hess}(h) = \frac{1}{\sqrt{1 + \left(\frac{x^2}{r^2 - x^2} \right)}} \begin{bmatrix} \frac{-r^2}{\sqrt{r^2 - x^2}^3} & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \frac{-r}{r^2 - x^2} & 0 \\ 0 & 0 \end{bmatrix} [g_{ij}]^{-1} \quad = \begin{bmatrix} \frac{r^2 - x^2}{r^2} & 0\,0 \quad 1 \end{bmatrix} \tag{2.40}$$

$$\implies \widehat{\mathsf{L}} = [h_{ij}][g_{ij}]^{-1} = \begin{bmatrix} \frac{-r}{r^2 - x^2} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{r^2 - x^2}{r^2} & 0\,0 \quad 1 \end{bmatrix} \tag{2.41}$$

$$= \begin{bmatrix} -\frac{1}{r} & 0 \\ 0 & 0 \end{bmatrix} \tag{2.42}$$

We see that $u_2 = (0,1)$ and $u_1 = (1,0)$ are eigenvectors for $\widehat{\mathsf{L}}$ with respective eigenvalues $\kappa_2 = -\frac{1}{r}, \kappa_1 = 0$. The Theorem of Olinde Rodriguez then suggests that u_2 points in the direction of maximum curvature of the surface, $-\frac{1}{r}$, which is predictably in the direction directly perpendicular to the trough, whereas the direction of least curvature is along the trough and is 0. The Theorem of Meusnier, Theorem 2.2, suggests that the normal curvature $\kappa_2 = -\frac{1}{r}$ is reasonable–any curve on the trough perpendicular to the ridge should have the curvature of a circle, and the negative simply indicates that we are on the "outside" of the surface. Finally, we note that at the ridge of the trough is exactly where $\nabla f = 0$, and the Weingarten map is exactly the Hessian matrix there.

2.3. Orthogonality of Principal Directions and Approximated Principal Directions

Our final point is to consider the relationship between the principal directions of a surface at a point. Viewing the surface in $\mathbb{R}^3$, we define the Hessian $\text{Hess}(x, y)$ of the surface L at a point (x, y) on the surface as the matrix of its second partial derivatives:

$$\text{Hess}(x, y) = \begin{bmatrix} L_{xx}(x, y) & L_{xy}(x, y) \\ L_{yx}(x, y) & L_{yy}(x, y) \end{bmatrix} \tag{2.43}$$

At any point (x, y) we denote the two eigenpairs of the Hessian matrix of h as

$$\text{Hess}(x, y)u_i = \kappa_i u_i \,, \quad i = 1, 2 \tag{2.44}$$

where κ_i and u_i are known as the *approximated principal curvatures* and *approximated principal directions* of $L(x, y)$, respectively, and we label such that $|\kappa_2| \geq |\kappa_1|$. Notably, $\text{Hess}(x, y)$ is a real, symmetric matrix (since $L_{xy} = L_{yx}$ and L is a real function) and thus its eigenvalues are real and its eigenvectors are orthonormal to each other, as given by following basic result from linear algebra [11].

Lemma 2.9. *Let A be a real, symmetric matrix. The eigenvalues of A are real and its eigenvectors are orthonormal to each other.*

Proof. Let $x \neq 0$ so that $Ax = \lambda x$. Then

$$\begin{aligned} \|Ax\|_2^2 = \langle Ax, Ax \rangle &= (Ax)^* Ax \\ &= x^* A^* Ax = x^* A^T Ax = x * AAx \\ &= x^* A \lambda x = \lambda x^* Ax \\ &= \lambda x^* \lambda x = \lambda^2 x^* x = \lambda^2 \|x\|_2^2 \end{aligned}$$

Upon rearrangement, we have $\lambda^2 = \frac{\|Ax\|_2^2}{\|x\|_2^2} \geq 0 \implies \lambda$ is real.

To prove that a set of orthonormalizable eigenvectors exists, let A be real, symmetric as above and consider the eigenpairs $Av_1 = \lambda_1 v_1$, $Av_2 = \lambda_2 v_2$ with $v_1, v_2 \neq 0$.

5 In the case that $\lambda_1 \neq \lambda_2$, we have

$$\begin{aligned}(\lambda_1 - \lambda_2)v_1^T v_2 &= \lambda_1 v_1^T v_2 - \lambda_2 v_1^T v_2 \\ &= (\lambda_1 v_1)^T v_2 - v_1^T(\lambda_2 v_2) \\ &= (Av_1)^T v_2 - v_1^T(Av_2) \\ &= v_1^T A^T v_2 - v_1^T A v_2 \\ &= v_1^T A v_2 - v_1^T A v_2 = 0\end{aligned}$$

Since $\lambda_1 \neq \lambda_2$, we conclude that $v_1^T v_2 = 0$.

In the case that $\lambda_1 = \lambda_2 =: \lambda$, we can define (as in Gram-Schmidt orthogonalization) $u = v_2 - \frac{v_1^T v_2}{v_1^T v_1} v_1$. This is an eigenvector for $\lambda = \lambda_2$, as

$$\begin{aligned}Au &= A\left(v_2 - \frac{v_1^T v_2}{v_1^T v_1} v_1\right) \\ &= Av_2 - \frac{v_1^T v_2}{v_1^T v_1} Av_1 \\ &= \lambda v_2 - \frac{v_1^T v_2}{v_1^T v_1} \lambda v_1 \\ &= \lambda\left(v_2 - \frac{v_1^T v_2}{v_1^T v_1} v_1\right) = \lambda u\end{aligned}$$

and is perpendicular to v_1, since

$$\begin{aligned} v_1^T u &= v_1^T \left(v_2 - \frac{v_1^T v_2}{v_1^T v_1} v_1 \right) \\ &= v_1^T v_2 - \left(\frac{v_1^T v_2}{v_1^T v_1} \right) v_1^T v_1 \\ &= v_1^T v_2 - v_1^T v_2 (1) = 0. \end{aligned}$$

□

Thus we see that the two principal directions form an orthonormal frame at each point (x,y) within the continuous image $L(x, y)$.

CHAPTER 3

THE UNISCALE FRANGI FILTER

We now seek to harness the ideas of this section to the task at hand: identifying curvilinear content within images.

The Frangi filter, first described by Alejandro Frangi et al. in [12] is a widely used Hessian-based filter within image processing. Hessian-based filters make use of the logical "proximity" of the Hessian to notions of curvature of surfaces, as developed in Section 2.2. Several such Hessian-based filters exist–see [13] and [14], as well as a comparison given in [15]. These filters use information about the principal curvatures, approximated as eigenvalues of the Hessian) at each point in the image to identify regions of significant curvature within an image.

Frangi's filter was orignally developed for vascular segmentation in images such as MRIs and it excels in that context.

The procedure for a single scale in a 2D image is as follows: Let λ_1, λ_2 be the two eigenvalues of the Hessian of the image at point (x, y), ordered such that $|\lambda_1| \leq |\lambda_2|$, and define the Frangi vesselness measure as:

$$\mathcal{V}_\sigma(x_0, y_0) = \begin{cases} 0 & \text{if} \quad \lambda_2 > 0 \\ \exp\left\{-\frac{A^2}{2\beta^2}\right\}\left(1 - \exp\left(-\frac{S^2}{2c^2}\right)\right) & \text{otherwise} \end{cases} \tag{3.1}$$

where

$$A := |\lambda_1/\lambda_2| \quad \text{and} \quad S := \sqrt{\lambda_1^2 + \lambda_2^2} \tag{3.2}$$

and β and c are tuning parameters. Before we discuss appropriate values for β and c, we first seek to highlight the significance of Eq. (3.1), and in particular, the ratios defined in Eq. (3.2). A and S

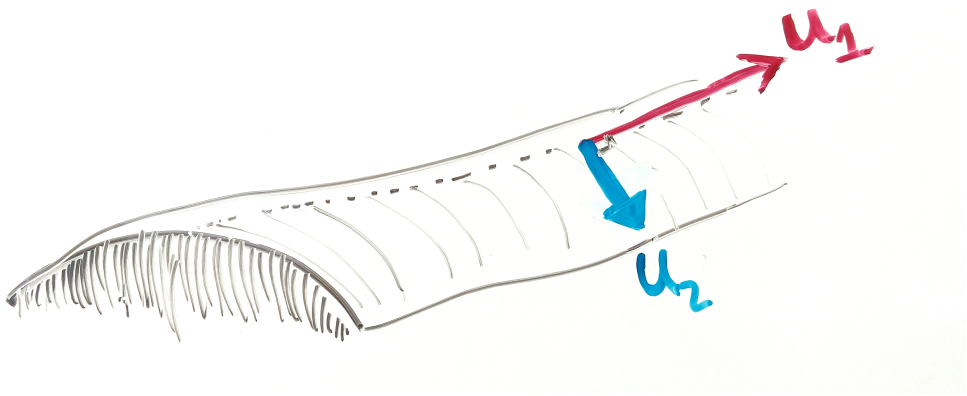


FIGURE 3: The principal eigenvectors at a ridge like structure

are known as the anisotropy measure and structureness measure, respectively. Consequently, we'll refer to the two factors in Eq. (3.1) as the anisotropy factor and structureness factor, respectively.

3.1. Anisotropy Measure

The anisotropy (or directionality) measure A is simply the ratio of magnitudes of λ_1 and λ_2. Since at a ridge point of a tubular structure, we should have $\lambda_1 \approx 0$ and $|\lambda_2| \gg |\lambda_1|$, a very small value of A would be present at a ridge of a tubular structure.

In Fig. 3, this situation is demonstrated. Here, u_1, u_2 form the orthogonal set of Hessian eigenvectors with corresponding eigenvalues λ_1 and λ_2. At such a ridgelike structure, we could predict the largest change in curvature to be straight down the ridge (in the direction of u_2), and the direction of least curvature to be directly along the ridge (in the direction of u_1). $\lambda_1 \approx 0$ and λ_2 is large and negative.

Of course, if the the ridge is perfectly circular along its cross section (as was in Section 2.2.4, it is of course apparent that λ_2 would be the same value at any place along the ridge (not just at its crest), and λ_1 would likewise be 0 at any such point. One could also imagine a similar situation in which the dropoff from crest to bottom gets increasingly steep. In such a case, λ_2 as a function of transverse position would in fact be largest nearest to the bottom. This thought experiment should dispel a naïve misunderstanding of the power of a Frangi filter: a high

anisotropy measure (and a large structureness measure) will not in general identify the crests of a ridge-like structure–it only will highlight that such a pixel is on a ridge-like structure at all. Thus, the anisotropy measure will not necessarily be at a maximum at the crest of the ridge, but instead, somewhere along it.

Similiarly, the vessel we we wish to identify can not be reasonably expected to behave as perfectly as our toy example. There will likely be small aberrations in a ridgelike structure, such as small divots or depressions in an overall ridge-like structure. Of importance in our data set later (Section 7.1), there will be points where we seem to "lose" our ridgelike structure, but this is simply due to an error in the sample.

Importantly, this formulation does not require λ_1 to be approximately zero, just that the curvature in the downward direction is much more significant than in the transverse direction.

Also the crest could be really flat ("hangar shaped"), in which case both are around zero. At the crest of the ridge, we would actually expect both u_1 and u_2 to be around 0, whereas a point somewhere between the crest and the "foot" of the ridge to contain the maximum u_2. We will fix this issue specifically by casting this as a multiscale problem in Chapter 5.

Two other ideas that could fix some other discrepancies mentioned above is to identify these ridges on their own, or also where the 'feet are'. We will discuss these ideas in Chapter 9.

3.2. Structureness measure

There is another concern with using the pure ratio $S := |\lambda_1/\lambda_2|$ as an identifying feature of ridgelike structures apart from the ones listed above. We could have $|\lambda_2| \gg |\lambda_1|$ in a relative sense, but still have $\lambda_2 \approx 0$. As a rather extreme example, we should certainly wish to differentiate a point on the surface where $\lambda_2 \approx 10^{-5}$ and $\lambda_1 \approx 10^{-10}$ from another point where $\lambda_2 \approx 10000$ and $\lambda_2 = 0.1$.

A natural fix to differentiate these points is to introduce a "structureness" measure to insure that there is in fact significant curvilinear activity at the point in question. Frangi used $S := \sqrt{(\lambda_1)^2 + (\lambda_2)^2}$, which is in fact the Frobenius norm of the Hessian matrix. Thus the Frangi filter should also prefer areas of great curvilinear content in the image first of all.

3.3. The Frangi vesselness measure

Our goal then is to attach a numerical measure to each pixel in the image (at a particular scale σ) that is large when the anisotropy measure A and the structureness measure S is sufficiently large.

The form Frangi arrived at in Eq. (3.1) in which a factor of $\exp(\cdots)$ and $(1-\exp(\cdots))$ are multiplied together are simply to ensure that the final vesselness measure V is largest when A is small and S is large enough, with rapid decay in other situations.

Frangi further strengthened the filter by adding an additional case to Eq. (3.1), ensuring that λ_2 is not positive. If we are indeed at a curvilinear ridge, we need the second derivative of the surface in the maximal direction to be negative, which hasn't been accounted for as yet in our formulation of A and S – we wish (for our purposes) to only identify when we are finding crests. A will still be small and S will still be large however if we identify a "trough".

The only perceivable difference is that the maximum normal curvature will be positive–we are at a local minimum in the direction of u_2. In situations where we wish to only identify ridges (as is the case here) we simply exclude any points where there is not a negative curvature in the maximal direction. Conversely, we could only seek to find valley, or local minima, as thus require $\lambda_2 > 0$, and set the vesselness measure to zero when $\lambda_2 < 0$.

3.4. Choosing parameters β and c

The parameters β and c are meant to scale so that the peaks of the anisotropy factor $\exp(\frac{-A^2}{2\beta^2})$ and the structureness factor $(1-\exp\left(\frac{-S^2}{2c^2}\right))$ coincide enough to be statistically significant at highly curvilinear structures, but rapidly decay in areas not associated with curvilinear content. What values of these parameters are appropriate is ultimately dependent on the context of the problem.

Frangi suggested for c that half of (the Frobenius norm of the) Hessian matrix is appropriate, simply because the minimum value of S is zero, and its maximum value is exactly the max Frobenius norm. With this in mind we would like to introduce the scaling factor γ, so that

$c = \gamma S_{\max}$. This creates a minor annoyance though: although the anisotropy factor can certainly attain a value of 1, if c is to take this "appropriate" value, the maximum value of the structureness factor is somewhat smaller than 1. In fact,

$$\begin{aligned}\max\{\mathcal{V}_\sigma\} &\leq \max\left(\exp\left(\frac{-A^2}{2\beta^2}\right)\right)\max\left(\left(1-\exp\left(\frac{-S^2}{2(\gamma S_{\max})^2}\right)\right)\right)\\ &\leq \max\left\{\left(1-\exp\left(\frac{-S^2}{2(\gamma S_{\max})^2}\right)\right)\right\}\\ &= \left(1-\exp\left(\frac{-(S_{\max})^2}{2(\gamma S_{\max})^2}\right)\right) = \left(1-\exp\left(\frac{-1}{2\gamma^2}\right)\right)\end{aligned} \tag{3.3}$$

Thus, when γ takes the suggested value of $\gamma = 1/2$, the above calculation suggests that the maximum theoretical value that the Frangi filter could attain at any scale is $\max\{\mathcal{V}_\sigma\} \leq 1-\exp(-1) \approx .8647$. This (among other obvious reasons) certainly justifies Frangi's description of the vesselness measure as only "probability-like." Still, we would like the filter's sensitivity to relative structureness to not have the effect of dampening the Filter as a whole, so we will introduce a rescaling factor a_γ, which is a explicit function of γ that rescales $\mathcal{V}_\sigma$ so that the structureness factor has a maximum output score of 1 regardless of choice of γ. Our modified Frangi vesselness measure is thus

$$\mathcal{V}_\sigma(x_0, y_0) = \begin{cases} 0 & \text{if} \quad \lambda_2 > 0 \\ a_\gamma \exp\left\{\frac{-A^2}{2\beta^2}\right\}\left(1-\exp\left\{\frac{-S^2}{2(\gamma S_{\max})^2}\right\}\right) & \text{otherwise} \end{cases} \tag{3.4}$$

where, as before,

$$A := |\lambda_1/\lambda_2|\,,\ S := \sqrt{\lambda_1^2+\lambda_2^2} \text{ and } a_\gamma = \left(1-\exp\left(\frac{-1}{2\gamma^2}\right)\right)^{-1}$$

and

$$|\lambda_1| \leq |\lambda_2| \text{ are eigenvalues of } \mathrm{Hess}_\sigma(\mathtt{I}(x_0, y_0))$$

For β, Frangi suggested an innocuous intermediate point, $\beta = 1/2$ (and thus $2\beta^2 = 1/2$).

As we will show later, choosing the structureness parameter γ is rather important for the context especially if the background (non-ridgelike structure) is significant and noisy. β should be strengthened/relaxed depending on how "flat" the ridgelike structure is. We shall show empirically that contexts in which more 'bloblike' structures are known to be present than that for which the Frangi filter was originally designed, we will benefit from a smaller choice of β.

Considering as the anisotropy measure $|\lambda_1/\lambda_2| \in [0, 1]$ (simply since $|\lambda_1| \leq |\lambda_2|$), we can actually visualize how much the anisotropy factor varies depending on our choice of β, as seen in Fig. 4.

We can theoretically choose any values $0 < \beta, \gamma < \infty$ for the two parameters. Two particular limits are of theoretical interest: as $\beta \to \infty$, the anisotropy factor tends to 1, and as $\gamma \to 0$, the structureness factor tends to 1, make the measure entirely irrelevant. Of course, in the limit that $\beta \to 0$ or $\gamma \to \infty$, their respective factors become 0, making the entire filter irrelevant.

We make a similar presentation of the dependence of the structureness kernel on its parameter γ, as you can see in Fig. 5.

The ultimate choice of these parameters β and γ have the overall effect of tuning the selectivity of the Frangi filter itself. In the figures in Appendix B, we plot the Frangi vesselness measure itself as a function of S and A while sweeping through different choices of β and γ.

We now take a quick tangent from our description of the Frangi filter to develop and justify our "multiscale" approach.

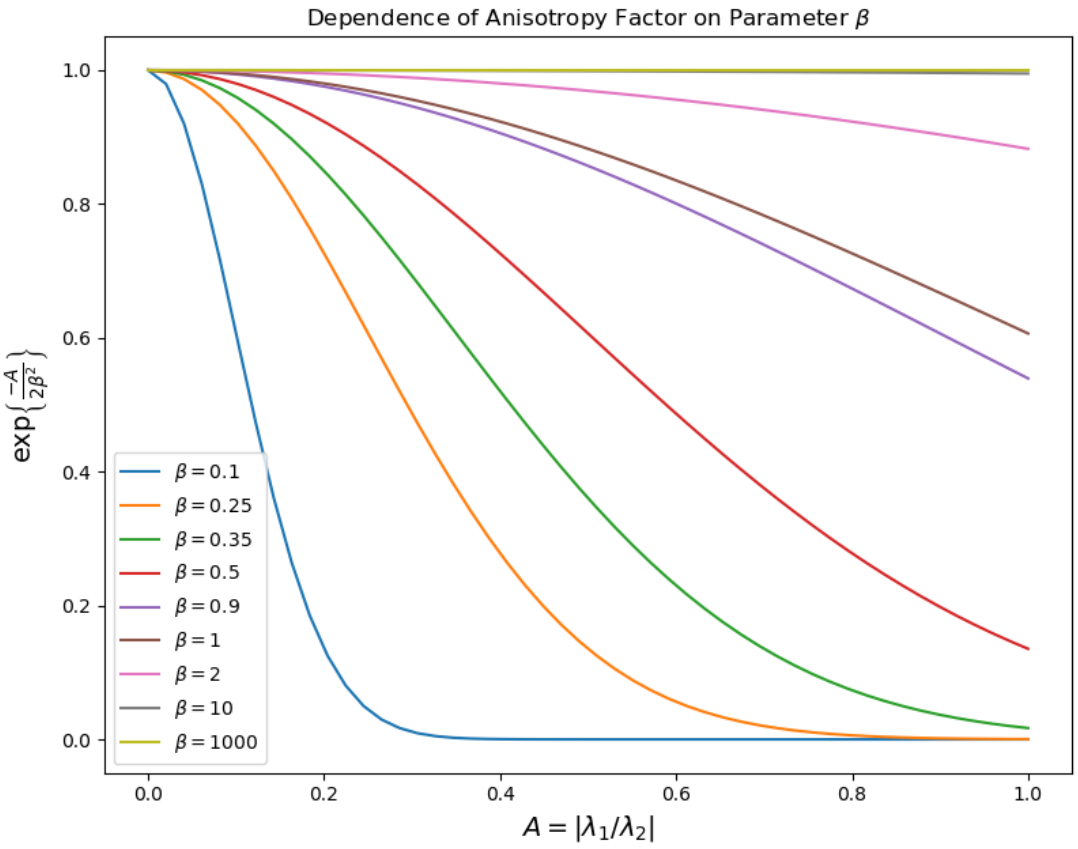


FIGURE 4: Dependence of the Anisotropy Factor on its Parameter

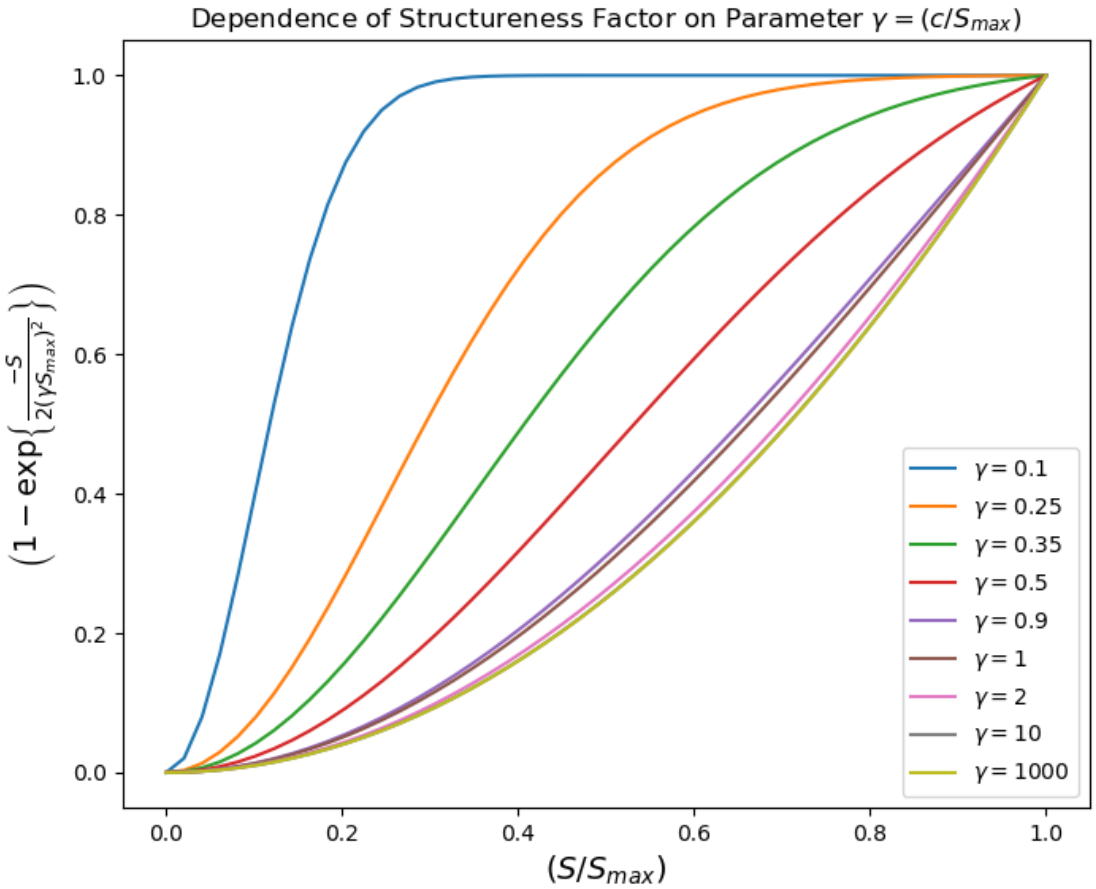


FIGURE 5: Dependence of the Structureness Factor on its Parameter

CHAPTER 4

LINEAR SCALE SPACE THEORY

Although the ideas presented above require differentiation of continuous surfaces, our image is in fact composed of discrete pixels. That is, our previous discussions have been in terms of an image as the continuous surface in Definition 2.2, rather than the more realistic discrete pixel matrix as in Definition 2.1. The present section seeks to address this disconnect. Divided differences can serve as an analogue of differentiation in discrete contexts, but our "derivative" and any use of it is then completely dependent on the bias of our limited sampling of the “true” 3D surface. Our main goal is to counter against some of the bias of our particular sampling. In particular, we wish to not over-represent structures that are clear at our resolution without giving appropriate weight to larger structures as well. Ideally, we would like statements we make about higher order phenomena to be stable enough at least that we would arrive at the same conclusions from analyzing a higher resolution image of the same surface. Koenderink [16] argued that "any image can be embedded in a one-parameter family of derived images, with resolution as the parameter in essentially only one unique way" given a few of the so-called scale space axioms. He, along with other independent efforts, showed that a small set of intuitive axioms imply that any such family of images $\{K_\sigma\}$ must satisfy the heat equation

$$\begin{cases} \Delta K(x,y;\sigma) = \dfrac{\partial K}{\partial \sigma}(x,y;\sigma) \text{ for } \sigma \geq 0 \\ K(x,y,0) = u_0(x,y) \end{cases} \tag{4.1}$$

where $u_0 : \mathbb{R}^2 \to \mathbb{R}$ is the original image (viewed as a continuous surface), σ is the resolution parameter, and $K : \mathbb{R}^2 \to \mathbb{R}$ for each fixed σ.

This result is intuitive and desirable–we would anticipate a lower-resolution version of our

original image to represent a diffusion or "blurring" of the initial scales information. Much work has been done to formalize this approach [17]. This has resulted in various formulations of a minimal set of axioms from which all other desirable properties of the scale space can be derived, culminating eventually in the necessity and sufficiency of Eq. (4.1) itself. We will refrain from such an exhaustive development in the present text, but rather list some desirable properties and outline the approach.

4.1. Properties/Axioms of Linear Scale Space Theory

To make matters manageable, we require the one-parameter family to be generated by an operator on the original image:

$$\{\, K(x,y;\sigma) = T_\sigma u_0 \mid \sigma \geq 0\,,\; K(x,y,;0) = u_0 \,\} \tag{4.2}$$

The following axioms are then requirements on what sort of operation T_σ should be.

Axiom 4.1 (Linear-shift and Rotational Invariance)**.** *We require that no position in the original signal is favored. This is intuitive, as our operation should apply to any image fairly, regardless of where content is found in the image, and cropping or rotating our initial image should not affect resolution of the content.*

Axiom 4.2 (Semigroup property)**.** *The semigroup property is simply that transforming the original image by some resolution σ should have the same overall effect of two successive transformations σ_1 and σ_2, i.e.*

$$T_\sigma u = T_{\sigma_1+\sigma_2} u \tag{4.3}$$

Axiom 4.3 (Continuity of Scale Parameter)**.** *There is no reason for the scale parameter to be discrete; we may alter the resolution with whatever precision we desire. That is, we take the resolution parameter σ to be any nonzero real number. Moreover, we require that the operator behaves continuously with respect to the scale parameter.*

The following requirement has great implication, and is also very successful in encoding our intuitive sense of resolution.

Axiom 4.4 (Causality Condition)**.** *The causality condition is the one that, as resolution decreases, no finer detail is introduced into the image. That is, as the scale increases, there will be no creation of local extrema that did not exist at a smaller scale.*

To make this more precise, if $K(x_0, y_0; \sigma_0)$ (that is, a point (x_0, y_0) at some particular resolution σ_0) is a local maximum at that resolution, then an increase in scale cannot make this maximum more prominent, i.e.

$$\begin{cases} \nabla K(x_0, y_0; \sigma_0) = 0 \\ \\ \Delta K(x_0, y_0; \sigma_0) < 0 \end{cases} \quad \Longrightarrow \quad K(x_0, y_0; \sigma_1) \leq K(x_0, y_0; \sigma_0) \; \forall \; \sigma_1 \geq \sigma_0 \tag{4.4}$$

Similarly, if $K(x_0, y_0; \sigma_0)$ is a local minimum (with respect to space), then an increase in scale cannot make such a valley more profound, i.e.

$$\begin{cases} \nabla K(x_0, y_0; \sigma_0) = 0 \\ \\ \Delta K(x_0, y_0; \sigma_0) > 0 \end{cases} \quad \Longrightarrow \quad K(x_0, y_0; \sigma_1) \geq K(x_0, y_0; \sigma_0) \; \forall \; \sigma_1 \geq \sigma_0 \tag{4.5}$$

This encodes our intuition that no image feature should be sharpened by a descrease in resolution. The only result is a (non-strictly) monotonic blurring of the image as scale parameter σ tends to infinity.

4.2. Sufficiency of the Gaussian Kernel

Although we will omit it here, the above requirements are actually sufficient in proving not only that the operator T_σ is a convolution, but that the heat equation described in Eq. (4.1) must hold. This has been shown in various ways, both by Koenderink [16], Babaud [18], as well as Lindeberg in [17]. Of course, once Eq. (4.1) has been established, it is straightforward to show that

$$K(x, y; \sigma) = T_\sigma u_0 = G_\sigma \star u_0 \quad \text{where} \quad G_\sigma := \frac{1}{2\pi\sigma^2} e^{\left(-|x|^2/(2\sigma^2)\right)} \tag{4.6}$$

is a solution. That is, the family can be generated by convolution with a Gaussian kernel. Lindeberg and others furthered this by arguing that the continuity of T_σ as σ approaches zero ultimately implies that convolution by a Gaussian is the *unique* operator that generates this scale space.

4.3. Scale Spaces over Discrete Structures

The above developments from scale space axioms have since been recast in terms of discrete structures, rather than continuous surfaces as above [19]. However, we've chosen to present the above in their original continuous surface for clarity of argument. The discrete case is not much different– we still have the same axioms, and it can be shown that the family of scaled images must simply satisfy a discrete version of the heat equation. However, viewing our actual image Definition 2.1 as a sample of a continuous surface Definition 2.2, we might expect our convolution by the Gaussian to "commute" with our supposed sampling of the continuous signal, or even that we could simply convolve our discrete signal with a discretely sampled Gaussian kernel. The latter in fact, seems to be an often implemented interpretation of scale space theory.

To be clear, the "sampled" 1D Gaussian Kernel we have in mind might be given by:

Definition 4.1 (Sampled Gaussian Kernel and Generated Family)**.**

$$g(n; \sigma) = \frac{1}{2\pi\sigma} e^{-n^2/2\sigma}, \quad -\infty < n < \infty$$

and the resulting (1D) convolution would be given by

$$K(x, \sigma) = \sum_{n=-\infty}^{\infty} g(n; \sigma) f(x - n) \quad \text{for} \quad x \in \mathbb{Z},\ \sigma > 0$$

In [19] and in particular [20], Lindeberg demonstrated that the sampled Gaussian kernel violates

not only semigroup property (Axiom 4.2), but–much less forgivably–the causality property (Axiom 4.2). There is absolutely no guarantee that convolution with a sampled Gaussian kernel will not create "spurious" structures as resolution increases.

Fortunately, Lindeberg was immediately able to remedy this by providing a discrete analogue of the Gaussian kernel, which does satisfy Axiom 4.4 and Axiom 4.2:

Definition 4.2 (Discrete Gaussian Kernel)**.** *The discrete Gaussian kernel, which can be shown to be a suitable generator for scale space, is given by*

$$T(n;\sigma) = e^{-\alpha\sigma} I_n(\alpha\sigma), \quad I_n(\sigma) = I_{-n}(\sigma) = (-1)^n J_n(i\sigma) \quad n \geq 0, \sigma, \alpha > 0 \tag{4.7}$$

where I_n are the modified Bessel functions of integer order based on the ordinary Bessel functions J_n, i.e.

$$I_n(x) = \sum_{m=0}^{\infty} \frac{1}{m!(m+n)!} \left(\frac{x}{2}\right)^{2m+n}, \quad n \geq 0$$

where we have taken the liberty of simplifying the typical definition [21] (which involves the gamma function), since we only desire Bessel functions of integer order. The parameter α above is simply an optional scaling parameter which is simply set to 1 hereforth.

The derived family of 1D signals is then given by

$$K(x,\sigma) = \sum_{n=-\infty}^{\infty} T(n;t) f(x-n) \quad \text{for} \quad x \in \mathbb{Z},\ t > 0 \tag{4.8}$$

The compatibility of scale space theory and derivatives on discrete structures and extension to two dimensions was also demonstrated by Lindeberg in [22] and [23]. In particular, we may take derivatives of the convolutions of our discrete images using, say, a local central difference. Lastly, the 2D version of the family given in Eq. (4.8) can be obtained by independent convolution of its dimensions (i.e. it is separable).

CHAPTER 5

THE MULTISCALE AND SIGNED FRANGI FILTER

With the notion of scale space established, we return to the task of computing the Frangi filter for discrete images. A central point of importance here is that certain ridgelike structures will be more prominent at different scales. Fig. 6 and Fig. 7 demonstrate this effect for two different placental samples. Here, a Frangi filter is applied to the sample at eight different scales ranging from small to large. The vesselness response is greater at smaller scales for smaller width vessels and smaller for larger width vessels. When the scale is large, the reverse is true: response to small vessels is nonexistent, and larger scales are prominent. A full treatment of applying the Frangi filter to these samples begins in Chapter 7.

5.1. The Multiscale Frangi Filter

Since our eventual goal is a complete extraction of the vasuclar network, a multiscale approach is the natural one. Considering the dependence of the Frangi filter's response on choice of scale as demonstrated in Chapter 3, we wish to probe at multiple scales regions that would receive a high vesselness score at any range and somehow merge the result.

Frangi [12] approached this problem by simply taking the maximum vesselness measure over all scales. Thus the multiscale Frangi vesselness score at the pixel (x_0, y_0) would be

$$\mathcal{V}_{\max}(x_0, y_0) := \max_{\sigma \in \Sigma} \{\mathcal{V}_\sigma(x_0, y_0)\} \tag{5.1}$$

where $\Sigma := \{\sigma_0, \sigma_1, \cdots, \sigma_n\}$ is the set of n scales at which to probe, and $\mathcal{V}_\sigma$ is the Frangi vesselness measure at scale σ for the pixel (x_0, y_0), as given in Eq. (3.4). Occasionally it will be useful to refer to the scale at which $\mathcal{V}_{\max}$ occurs for a particular pixel, so we analogously define

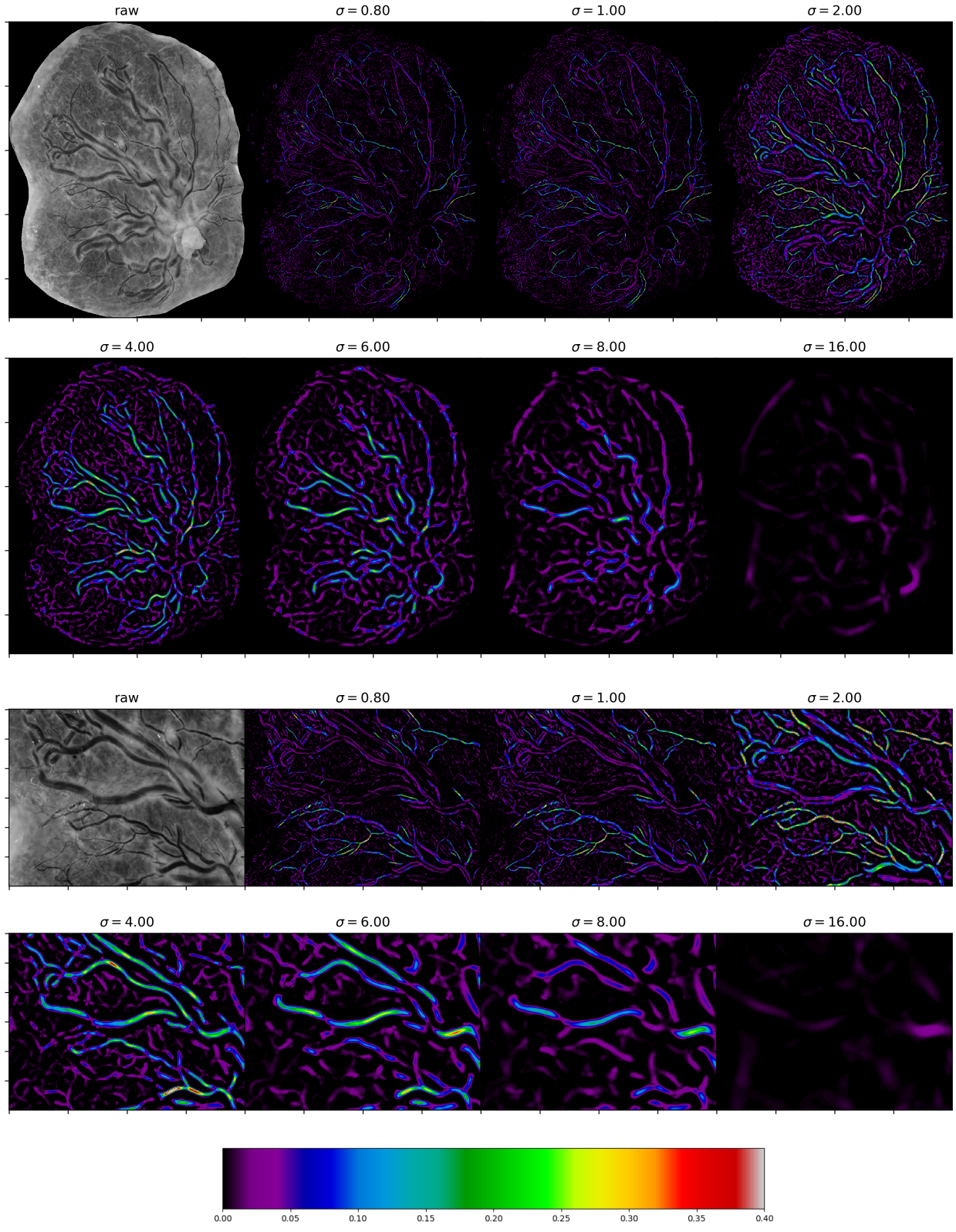


FIGURE 6: Example scalewise Frangi output (plate and inset) (Example 1)

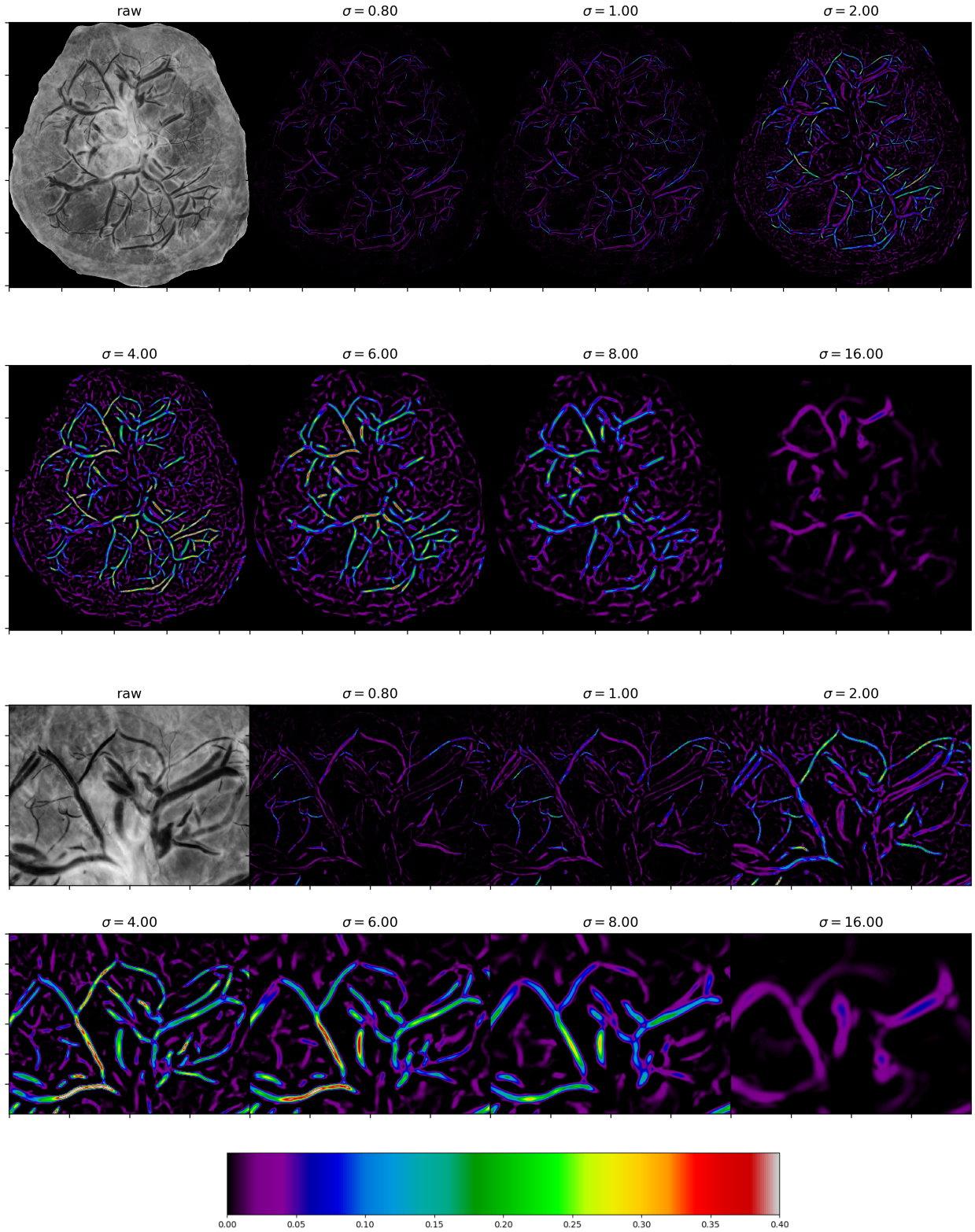


FIGURE 7: Example scalewise Frangi output (plate and inset) (Example 2)

$$\mathcal{V}_{\text{argmax}}(x_0, y_0) := \arg\max_{\sigma \in \Sigma} \{\mathcal{V}_\sigma(x_0, y_0)\} \tag{5.2}$$

The set of scales Σ should be chosen to be representative enough of all scales where meaningful content is expected to be found. Unfortunately this appropriate range of scales may not be known *a priori*. We will address this issue in Chapter 9.

Finally, we may also wish to deal with the unmerged vesselness measures as a whole, i.e. without taking a maximum at all. In this case, we will refer to $\mathcal{V}_\Sigma$, the 3D matrix of shape $n \times (M \times N)$, where M and N are the dimensions of the image matrix `I` as in Definition 2.1.

$$\mathcal{V}_\Sigma(x_0, y_0; \sigma) := \mathcal{V}_\sigma(x_0, y_0) \tag{5.3}$$

After the maximization in Eq. (5.1), we are left with a matrix with as many pixels as the original image, all with a vesselness measure between 0 and 1 for each pixel in the image.

At this point, Frangi [12] refrained from explicitly interpreting the score assigned by Eq. (5.1); that is–whether a particular pixel (x_0, y_0) in the image definitely respresents a vessel or not based on its Frangi score. Instead, he cautioned that the result should not be used as a segmentation method alone; moreover, the width of the vasculature cannot be determined rigorously from the Frangi filter, as discussed in Chapter 3.

Nonetheless, we wish to demonstrate the usefulness of the Frangi filter within our image domain towards segmentation. We should at least expect that a well-tuned Frangi filter on an appropriately registered and denoised sample should assign its highest scores to vessel pixels. We can select these strong Frangi responses and use them as seeds for some subsequent algorithm. We will discuss methods of harnessing the Frangi result to the task of segmentation in Chapter 9. A straightforward enough approach would be to simply threshold at some fixed value α. Such thresholding was used in [2].

5.2. The Signed Frangi Filter

We finally introduce the novel (yet straightforward) notion of the signed Frangi filter. As will be shown in Chapter 9, we can befefit from simultaneously calculate for a dark background and a light background. Since the Frangi filter normally throws away any response where $\lambda_2 < 0$ (if dark curvilinear features are targeted) or $\lambda_2 > 0$ (if light curvilinear features are targeted), we lose no computation time at all (although we must store more results). After computing the multiscale result, we can easily separate these into a positive and a negative strain, which we will denote $\mathcal{V}_{\text{max}}^{(+)}$ and $\mathcal{V}_{\text{max}}^{(-)}$. Our $\mathcal{V}_{\text{max}}^{(+)}$ is the same as our $\mathcal{V}_{\text{max}}$ before, and $\mathcal{V}_{\text{max}}^{(-)}$ is the same result as if we had taken the Frangi filter while only looking for the opposite type (light/dark) curvilinear feature, i.e. by changing the piecewise case of Eq. (3.1) to $\lambda_2 > 0$. Plotting $\mathcal{V}_{\text{max}}$ over a scale of $[-1, 1]$ demonstrates an interesting effect, as shown in Fig. 30. Whereas the Frangi filter generally is not reliable in terms of accurately predicting widths of trough-like (or ridge-like) curvilinear features, we *can* get a sense of the width of the vessel in our particular application by simulataneously considering $\mathcal{V}_{\text{max}}^{(+)}$ and $\mathcal{V}_{\text{max}}^{(-)}$. We will develop a method of utilizing this observation in Chapter 9.

All that remains to describe mathematically is how to actually calculate the derivatives of our images and deal with the ultimately discrete nature of our samples.

CHAPTER 6

FFT-BASED DISCRETE DERIVATIVES

According to Section 4.3, we may calculate derivatives of our structure by calculating a gradient on our convolved image. Our method of calculating the gradient of a matrix uses a second-order accurate central difference, as in [24].

We note in passing that we may take the derivative of the Gaussian kernel and then convolve it, and the effect will be the same as if we had taken the derivative subsequently [6]. This could offer some computational speedup if we wish to run this procedure on many samples and fixed scale sizes. For the time being, we will convolve first, although our method will differ from standard convolution.

6.1. Fourier Transforms

In practice, the convolutions described above are very slow for large scales (σ), as the size of the kernel is very large. Instead, we will perform a fast Fourier transform, which requires only $\mathcal{O}\left(N \cdot \log_2 N\right)$ operations for a one dimension signal of length N, as compared to the N^2 operations required of a conventional discrete Fourier transform [6]. We will briefly outline the theory of Fourier transforms.

6.1.1. Fourier Transform of a Continuous 1D Signal

A periodic signal (real valued function) $f(t)$ of period T can be expanded in an infinite basis as follows:

$$f(t) = \sum_{-\infty}^{\infty} c_n e^{i\frac{2\pi n}{T}t}, \quad c_n = \frac{1}{T}\int_{-T/2}^{T/2} f(t) e^{-i\frac{2\pi n}{T}t} dt \tag{6.1}$$

The Fourier transform of a 1D continuous function is defined by

$$F(\mu) := \mathcal{F}\{f(t)\} = \int_{-\infty}^{\infty} f(t)e^{i2\pi\mu}dt \tag{6.2}$$

An inverse transform will then recover our original signal:

$$f(t) = \mathcal{F}^{-1}\{F(\mu)\} = \int_{-\infty}^{\infty} F(\mu)e^{i2\pi\mu t}dt \tag{6.3}$$

Together, Eq. (6.2) and Eq. (6.3) are referred to as the *Fourier transform pair* of the signal $f(t)$.

6.1.2. Fourier Transform of a Discrete 1D signal

We wish to develop the Fourier transform pair for a discrete signal., following [6]. We frame the situation as follows: a continuous function $f(t)$ is represented as the sampled function $\tilde{f}(t)$ by multiplying it by a sampling (or impulse) function, an infinite series of discrete impulses with equal spacing ΔT:

$$s_{\Delta T}(t) := \sum_{n=-\infty}^{\infty} \delta[t - n\Delta T], \quad \delta[t] = \begin{cases} 1, & t = 0 \\ 0, & t \neq 0 \end{cases} \tag{6.4}$$

where $\delta[t]$ is the discrete unit impulse.

The discrete sample $f(t)$ is then constructed from $f(t)$ by

$$\tilde{f}(t) = f(t)s_{\Delta T}(t) \tag{6.5}$$

From this we can calculate $\tilde{F}(t)$. Given the discrete signal $\tilde{f}$, we construct the transform $\tilde{F}(\mu) = \mathcal{F}\{\tilde{f}(t)\}$. by expanding $\tilde{f}$ in the same infinite basis as the continuous case.

$$\tilde{F}(\mu) = \sum_{n=-\infty}^{\infty} f_n e^{-i2\pi\mu n\Delta T}, \quad f_n = \tilde{f}(n) = f(n\Delta T) \tag{6.6}$$

The transform is a continuous function with period $1/\Delta T$.

6.1.3. 2D DFT Convolution Theorem

Theorem 6.1 (2D DFT Convolution Theorem)**.** *Given two discrete functions are sequences with the same length. $f(x,y)$ and $h(x,y)$ for integers $0 < x < M$ and $0 < y < N$, we can take the discrete fourier transform (DFT) of each, where $\mathcal{D}\{\cdots\}$ denotes the DFT.*

$$F(u,v) := \mathcal{D}\{f(x,y)\} = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-2\pi i\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{6.7}$$

$$H(u,v) := \mathcal{D}\{h(x,y)\} = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} h(x,y)e^{-2\pi i\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{6.8}$$

and given the convolution of the two functions

$$(f \star h)(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)h(x-m,y-n) \tag{6.9}$$

then $(f \star h)(x,y)$ and $MN \cdot F(u,v)H(u,v)$ are transform pairs, i.e.

$$(f \star h)(x,y) = \mathcal{D}^{-1}\{MN \cdot F(u,v)H(u,v)\} \tag{6.10}$$

The proof follows from the definition of convolution, substituting in the inverse-DFT of f and h, and then rearrangement of finite sums.

Proof.

$$(f \star h)(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)h(x-m,y-n) \tag{6.11}$$

$$= \sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left(\sum_{p=0}^{M-1}\sum_{q=0}^{N-1} F(p,q)e^{2\pi i\left(\frac{mp}{M}+\frac{nq}{N}\right)}\right)\left(\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} H(u,v)e^{2\pi i\left(\frac{u(x-m)}{M}+\frac{v(y-n)}{N}\right)}\right) \tag{6.12}$$

$$= \left(\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} H(u,v)e^{2\pi i\left(\frac{ux}{M}+\frac{vy}{N}\right)}\right)\left(\sum_{p=0}^{M-1}\sum_{q=0}^{N-1} F(p,q)\left(\sum_{m=0}^{M-1} e^{2\pi i\left(\frac{m(p-u)}{M}\right)}\right)\left(\sum_{n=0}^{N-1} e^{2\pi i\left(\frac{n(q-v)}{N}\right)}\right)\right) \tag{6.13}$$

$$= \left(\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} H(u,v)e^{2\pi i\left(\frac{ux}{M}+\frac{vy}{N}\right)}\right)\left(\sum_{p=0}^{M-1}\sum_{q=0}^{N-1} F(p,q)\left(M\cdot\hat{\delta}_M(p-u)\right)\left(N\cdot\hat{\delta}_M(q-v)\right)\right) \tag{6.14}$$

$$= \left(\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} H(u,v)e^{2\pi i\left(\frac{ux}{M}+\frac{vy}{N}\right)}\right)\cdot MNF(u,v) \tag{6.15}$$

$$= MN\cdot\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)H(u,v)e^{2\pi i\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{6.16}$$

$$= MN\cdot\mathcal{D}^{-1}\{FH\} \tag{6.17}$$

where

$$\hat{\delta}_N(k) = \begin{cases} 1 & \text{when } k = 0 \mod N \\ 0 & \text{else} \end{cases} \tag{6.18}$$

□

Above, we make use of the following lemma

Lemma 6.2. *Let j and k be integers and let N be a positive integer. Then*

$$\sum_{n=0}^{N-1} e^{2\pi i\left(\frac{n(j-k)}{N}\right)} = N\cdot\hat{\delta}_N(j-k) \tag{6.19}$$

Proof. Consider the complex number $e^{2\pi i(j-k)/N}$. Note first that this is an N-th root of unity, since

$$\left(e^{2\pi i(j-k)/N}\right)^N = e^{2\pi i(j-k)} = \left(e^{2\pi i}\right)^{(j-k)} = 1^{(j-k)} = 1$$

In other words, $e^{2\pi in(j-k)/N}$ is a root of $z^N - 1 = 0$, which we can factor as

$$z^N - 1 \;=\; (z-1)\left(z^{n-1} + \cdots + z + 1\right) \;=\; (z-1)\sum_{n=0}^{N-1} z^n. \tag{6.20}$$

thus giving us

$$0 = \left(e^{2\pi i(j-k)/N} - 1\right)\sum_{n=0}^{N-1} e^{2\pi in(j-k)/N} \tag{6.21}$$

To prove the claim in Eq. (6.19), we consider two cases: First, if $j-k$ is a multiple of N, we of course have $e^{2\pi in(j-k)/N} = \left(e^{2\pi i}\right)^{n(j-k)/N} = 1$ and thus the left side of Eq. (6.19) reduces to

$$\sum_{n=0}^{N-1}\left(e^{2\pi i}\right)^{n(j-k)/N} = \sum_{n=0}^{N-1}(1) = N$$

.

In the case that $j-k$ is *not* a multiple of N, we refer to Eq. (6.21). The first factor is not zero since, $\left(e^{2\pi i(j-k)/N}\right) \neq 1$ (simply since $(j-k)/N$ is not an integer), and thus it must be that the second factor is 0:

$$\sum_{n=0}^{N-1}\left(e^{2\pi i(j-k)/N}\right)^n = 0$$

We can combine these two cases by invoking the definition of Eq. (6.18), giving us the result. □

6.2. The Fast Fourier Transform

As noted, the above result applies to the Discrete Fourier Transform. We actually achieve a convolution speedup using a Fast Fourier Transform (FFT) instead. We follow the developments of [6]. For clarity, we present the following theorems which allow a framework to calculate a 2D

Fourier transforms quickly.

First, a 2D DFT may actually be calculated via two successive 1D DFTs, which can be seen through a basic rearrangement, as follows:

$$F(\mu,\nu) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-i2\pi(\mu x/M+\nu y/N)} \tag{6.22}$$

$$= \sum_{x=0}^{M-1} e^{-i2\pi\mu x/M}\left[\sum_{y=0}^{N-1} f(x,y)e^{-i2\pi\nu y/N}\right] \tag{6.23}$$

$$= \sum_{x=0}^{M-1} e^{-i2\pi\mu x/M}\,\mathcal{F}_x\{f(x,y)\} \tag{6.24}$$

$$= \mathcal{F}_y\{\mathcal{F}_x\{f(x,y)\}\} \tag{6.25}$$

where $\mathcal{F}_{x'}$ refers to the 1D discrete Fourier transform of the function with respect to the variable x' only.

Thus, to calculate the Fourier transform $F(u,v)$ at the point u,v requires the computation of the transform of length N for each iterated point $x \in 0,\cdots,M-1$. Thus there are MN complex multiplications and $(M-1)(N-1)$ complex additions in this sequence required for each point u,v that needs to be calculated. Overall, for all points that need to be calculated, the total order of calculations is on the order of $(MN)^2$ We'll also mention that the values of $e^{-i2\pi m/n}$ can be provided by a lookup table rather than ad-hoc calculation.

We now show that a considerable speedup can be achieved through elimination of redundant calculations. In particular, we wish to show that the calculation of a 1D DFT of signal length $M = 2^n, n \in \mathbb{Z}_+$ can be reduced to calculating two half-length transforms and an additional $M/2 = 2^{n-1}$ calculations.

To simplify our notation we will use a new notation for the Fourier kernels/basis functions. Let the 1D Fourier transform be given by

$$F(u) = \sum_{x=0}^{M-1} f(x) W_M^{ux}, \quad \text{where} \quad W_m := e^{-i2\pi/m} \tag{6.26}$$

We'll define $K \in \mathbb{Z}_+ : 2K = M = 2^n$ (i.e. $K = 2^{n-1}$).

We use this to rewrite the series in Eq. (6.26) and split it into odd and even entries in the summation

$$F(u) = \sum_{x=0}^{2K-1} f(x) W_{2K}^{ux} \tag{6.27}$$

$$= \sum_{x=0}^{K-1} f(2x) W_{2K}^{u(2x)} + \sum_{x=0}^{K-1} f(2x+1) W_{2K}^{u(2x+1)} \tag{6.28}$$

We'll get a few identities out of the way (where $m, n, x \in \mathbb{Z}_+$ arbitrary).

$$W_{(2m)}^{(2n)} = e^{\frac{-i2\pi(2m)}{2m}} = e^{\frac{-i2\pi m}{n}} = W_m^n \tag{6.29}$$

$$W_m^{(u+m)x} = e^{\frac{-i2\pi(u+m)x}{m}} = e^{\frac{-i2\pi unx}{m}} e^{\frac{-i2\pi mx}{m}} = e^{\frac{-i2\pi ux}{m}}(1) = W_m^{ux} \tag{6.30}$$

$$W_{2m}^{(u+m)} = e^{\frac{-i2\pi(u+m)}{2m}} = e^{\frac{-i2\pi ux}{2m}} e^{-i\pi} = W_{2m}^u e^{-i\pi} = -W_{2m}^u \tag{6.31}$$

Thus we can rewrite Eq. (6.28) as

$$F(u) = \sum_{x=0}^{K-1} f(2x) W_{2K}^{2ux} + \sum_{x=0}^{K-1} f(2x+1) W_{2K}^{2ux} W_{2K}^u \tag{6.32}$$

$$\implies \quad F(u) = \left(\sum_{x=0}^{K-1} f(2x) W_K^{ux} \right) + \left(\sum_{x=0}^{K-1} f(2x+1) W_K^{ux} \right) W_{2K}^u \tag{6.33}$$

The major advance comes via using the identities Eq. (6.29) to consider the Fourier

transform K frequencies later :

$$F(u+K) = \left(\sum_{x=0}^{K-1} f(2x) W_K^{(u+K)x}\right) + \left(\sum_{x=0}^{K-1} f(2x+1) W_K^{(u+K)x}\right) W_{2K}^{(u+K)} \tag{6.34}$$

$$\implies F(u+K) = \left(\sum_{x=0}^{K-1} f(2x) W_K^{ux}\right) - \left(\sum_{x=0}^{K-1} f(2x+1) W_K^{ux}\right) W_K^u \tag{6.35}$$

Comparing Eq. (6.33) and Eq. (6.35), we see that the expressions within parentheses are identical. What's more, these parentetical expressions are functionally identical to discrete fourier transforms themselves. Let's notate them as follows:

$$\begin{aligned} \mathcal{D}_u\{f_{\text{even}}(t)\} &:= \sum_{x=0}^{K-1} f(2x) W_K^{ux} \\ \mathcal{D}_u\{f_{\text{odd}}(t)\} &:= \sum_{x=0}^{K-1} f(2x+1) W_K^{ux} \end{aligned} \tag{6.36}$$

If we're calculating an M point transform (i.e. we're wishing to calculate $F(1), ..., F(M)$), once we've calculated the first K discrete frequencies (i.e. $F(1), \cdots, F(K)$) we may simply reuse the two values we've calculated in Eq. (6.36) to calculate the next $F(K+1), .., F(K+K) = F(M)$. Since each expression in parentheses involves K complex multiplications and $K-1$ complex additions, we are effectively saving $K(2K-1)$ calculations in computing the entire spectrum $F(1), .., F(M)$. When M is large, the payoff is undeniable.

In fact, through counting calculations and then doing a proof by induction, we can show that the effective number of calculations is given by $M \log_2 M$.

Of course, since Eq. (6.36) are DFTs themselves, there's nothing stopping us from reiterating this procedure; if M is substantially large, we can just as easily repeat this process a few times.

Of course, our development was for $1D$. We can extend this to $2D$ by taking note of Eq. (6.22).

Finally we note the inverse DFT can actually be found via a DFT of the complex conjugate of the original signal, and of course we may translate that operation to a FFT.

6.3. Calculating the Hessian via FFT: A demonstration

Efficient implementation of the Frangi filter ultimately relies on performing a 2D Gaussian blur in frequency space. Here we demonstrate that our FFT implementation of Gaussian blur is commensurate with other implementations.

In Fig. 8, we demonstrate the compatibility of standard convolution and FFT convolve. Each row corresponds to a different scale at which Gaussian blurring occurs. Column (a) is standard convolution with a sampled Gaussian kernel, column (b) is FFT-convolution with a Gaussian kernel, and column (c) is a FFT-convolution with the "discrete Gaussian kernel". In column (d), the 1D discrete Gaussian kernel (in green) is plotted against the sampled continuous Gaussian kernel (in black). Note that each of the images in the first three columns are scaled the same.

We also demonstrated the "semigroup property" of Gaussian kernel convolution. For a large scale ($\sigma = 45$) Gaussian blur ((a) - standard convolution with sampled kernel, (b) FFT with sampled kernel, (c) FFT with discrete kernel), the top row is one round of Gaussian blur with $\sigma = 45$ and the bottom row is two progressive passes of Gaussian blur ($\sigma_1 = 10, \sigma_2 = 35$). The mean squared error and mean absolute error between the one-pass and two-pass versions are outputted in Table 1. Code for this demo can be found in `hfft.semigroup_demo`. The discrete kernel performs very slightly better than the sampled versions. We originally attempted this demonstration with a much larger sigma (say $\sigma = 150$) and multiple iterations, but unfortunately multiple passes cause the "noise" from zeroing out around the boundaries to become very noticable after several iterations. Here, we've opted to crop out a radius of pixels from around the edges equal to the standard deviation of the Gaussian before we calculated the MAE or MSE, to reduce noise from the border.

We further confirm the commensurate nature of Gaussian blur techniques by comparing

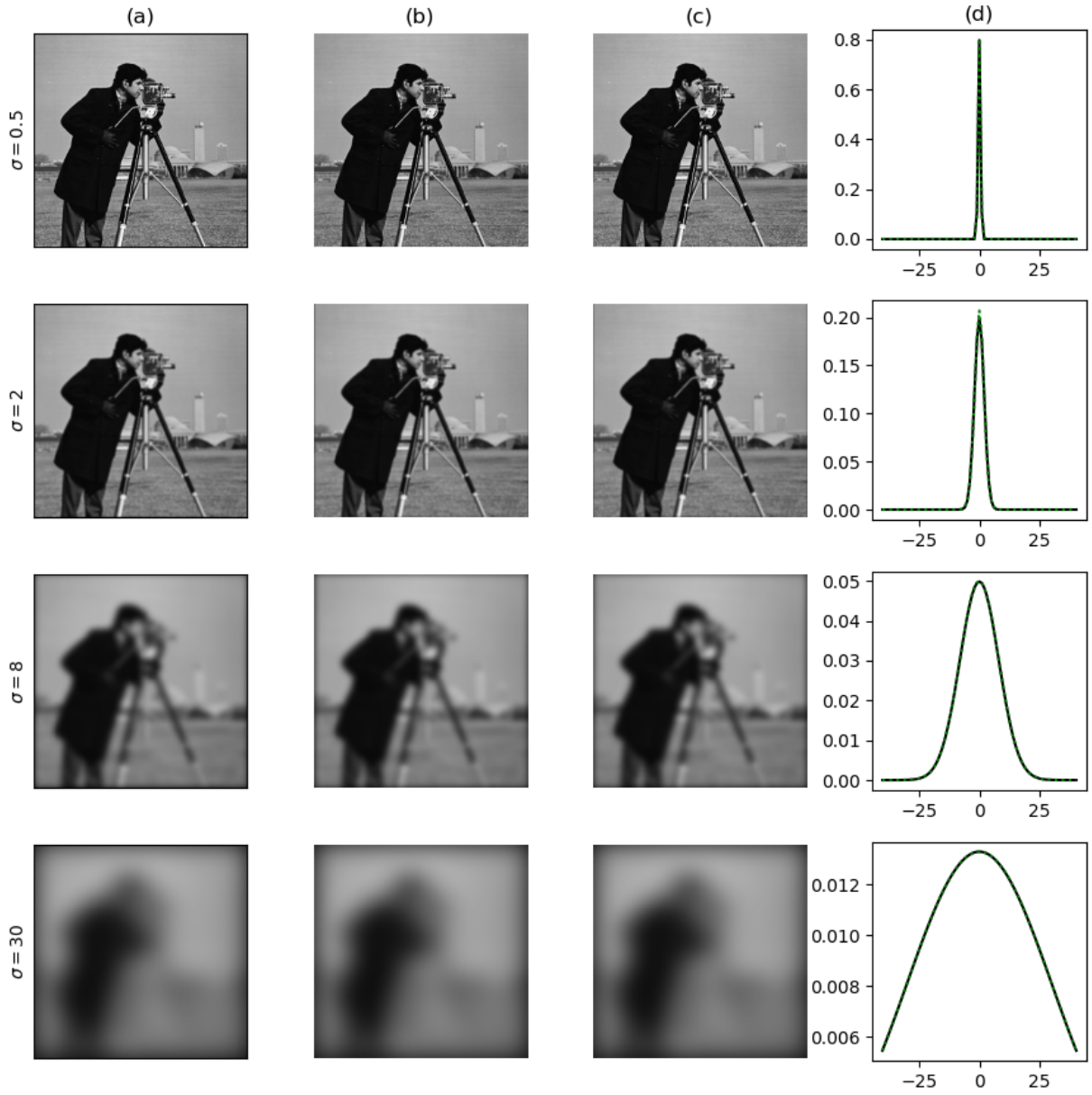


FIGURE 8: Compatibility of Gaussian convolution strategies

convolution implementation	MSE	MAE
spatial convolution, sampled kernel (A)	0.00054426	0.02015643
FFT convolution, sampled kernel (B)	0.00055205	0.02029916
FFT convolution, discrete kernel (C)	0.00054406	0.02015336

TABLE 1: MSE and MAE of one-pass vs. two pass Gaussian blur

	A	B	C
A	-	1.296e-03	6.772e-06
B	-	-	1.247e-03
C	-	-	-

TABLE 2: MSE of Gaussian blurs ($\sigma = 0.3$)

	A	B	C
A	-	4.256e-06	5.537e-08
B	-	-	4.337e-06
C	-	-	-

TABLE 3: MSE of Frangi scores $\sigma = 0.3$

	A	B	C
A	-	9.012e-06	8.629e-09
B	-	-	9.031e-06
C	-	-	-

TABLE 4: MSE of Gaussian blurs of an image ($\sigma = 5$)

	A	B	C
A	-	9.388e-05	8.383e-07
B	-	-	9.599e-05
C	-	-	-

TABLE 5: MSE of Frangi scores $\sigma = 5$

the three techniques on a placental image and using each to calculate Frangi targets. The code can be found in `hfft_accuracy.py`. In Tables 2 to 5 we compare the mean squared error of a single image blurred (A) with standard spatial convolution, (B) with FFT sampled Gaussian kernel, and (C) with the discrete kernel. We see that the standard convolution and discrete convolution are very similar, while the sampled discrete Gaussian is off by two orders of magnitude, but still reasonably small. We further confirm these by viewing the grayscale intensity of the image from 0 to 1 and the Frangi targets themselves across an arbitrarily chosen horizontal cross section of the image Fig. 9, the peaks of the Gaussian blurred image all still occur at the same places, as do the Frangi responses. We repeated this procedure up to $\sigma = 90$ and found a situation similar to $\sigma = 5$; it was only in very small scales where there was any noticeable difference at all.

Finally, we wish to demonstrate the point of this comparison–that $FFT-based$ convolution is much faster than spatial convolution. We took a much larger sample (2200×2561) and timed each method of convolution (average of three trials) for a large number of samples: logarithmic between $\sigma = 1$ and $\sigma = 128$ with 32 steps. We plot the result in Fig. 10. It shows that the convolution time seems to at least linearly increase with the size of the kernel, whereas FFT is independent of choice of scale. This is to be expected, as convolving with a Gaussian kernel in spatial coordinates requires a greater number of calculations as σ increases, whereas the size of

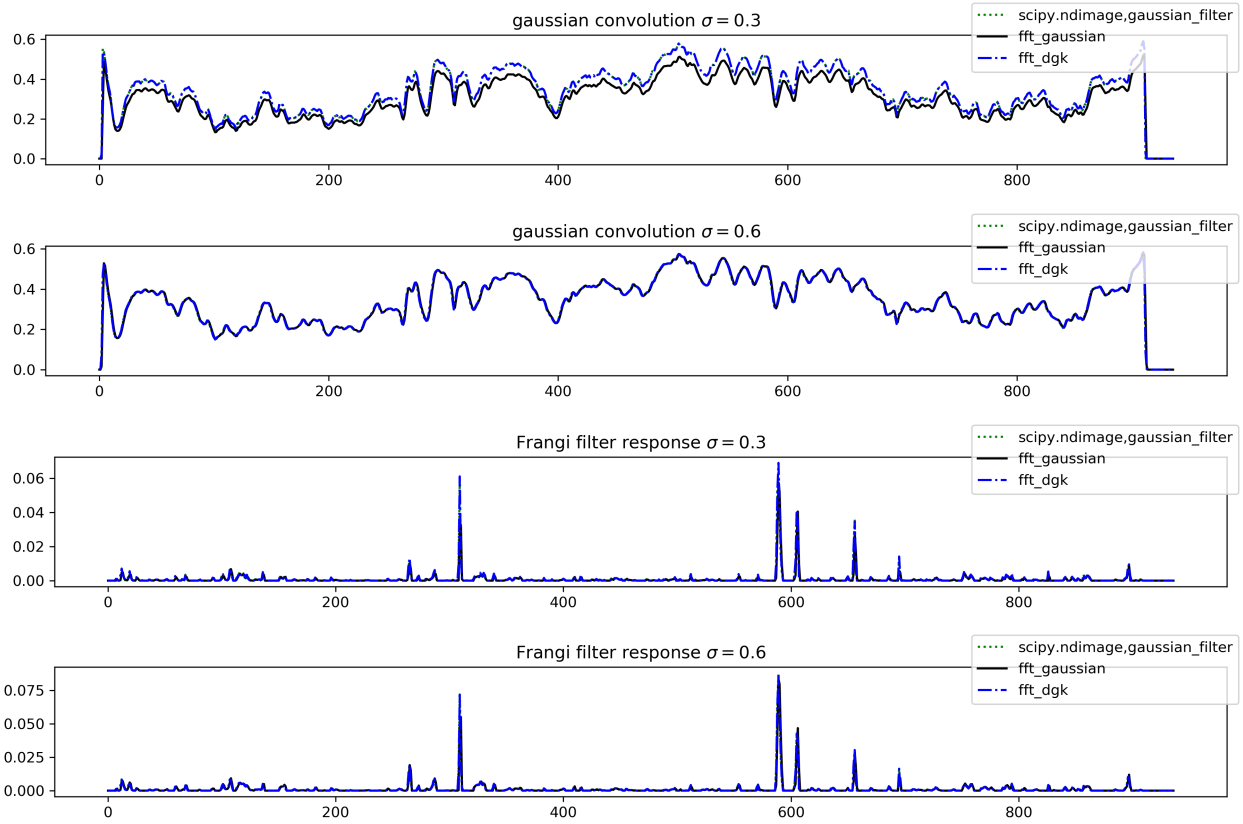


FIGURE 9: Image cross-section of Gaussian-blurred (grayscale) placental sample

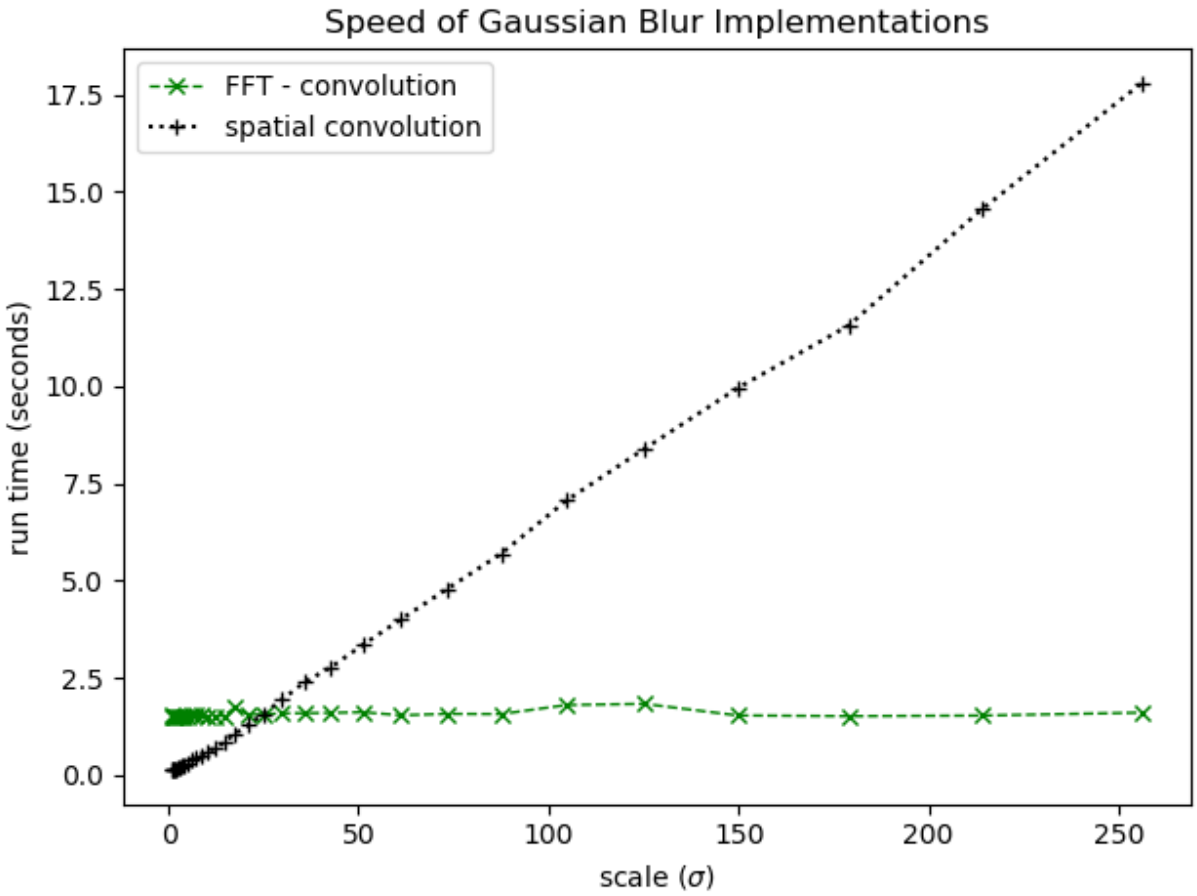


FIGURE 10: Runtime comparison of Gaussian convolution implementations

the kernel does not change in our frequency space convolution.

CHAPTER 7

RESEARCH PROTOCOL

7.1. Samples / Image Domain

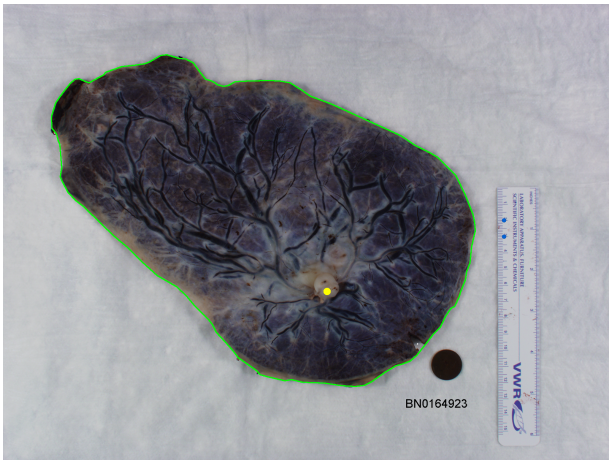
We ultimately perform a multiscale Frangi prefilter on a subset of 201 color images of placental samples from a private database provided by the National Children's Study, which had been prepared for a different study. A detailed description of the data set is given in [1]. A description of the cleaning and fixing procedure of the samples prior to their being photographed is given in [5]. The samples are provided as XCF files (the native project file for GIMP) and contain four major layers.

7.1.1. A representative sample

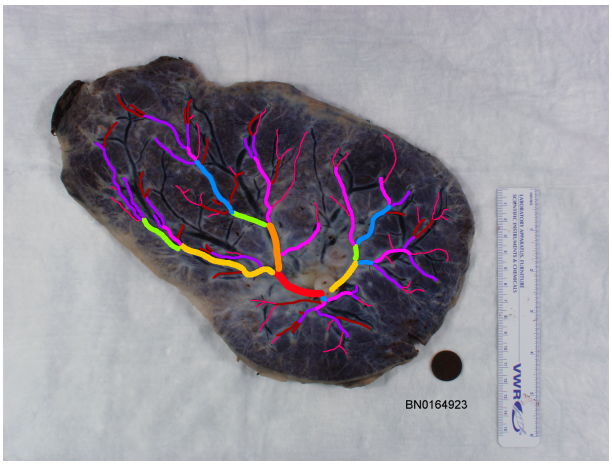
The layers together give a hand tracing of the vascular network and perimeter. A sample of overlaid layers in a representative sample (with ID number "BN0164923") is given in Fig. 11.

Each layer is roughly 1954x1200 pixels (with some occasional variation). In Fig. 11, we see these four layers of a characteristic sample. Fig. 11a is the base image. A cleaned, formalin-fixed placenta is placed on a table with a camera a fixed distance away, and a ruler and penny (presumably for redundancy) are placed nearby to aid registration and calibration of the resolution. The resolution of each sample is roughly 46 pixels per centimeter. Fig. 11b is a tracing (in green) of the perimeter of the placenta. The umbilical cord insertion point (UCIP) is notated in yellow. Two cyan marks are placed on consecutive centimeter markings on the ruler (the dots are enlarged and shown as a darker blue here for clarity). Fig. 11c and Fig. 11d are each hand traces of the PCSVN, with a layer for each the arteries and veins. These layers are simultaneously overlain on the base image in Fig. 11e. The coloration is meant to indicate the diameter of each vessel. The diameters are binned into 9 discrete widths, odd integers from 3 to 19 pixels. Vessels of smaller

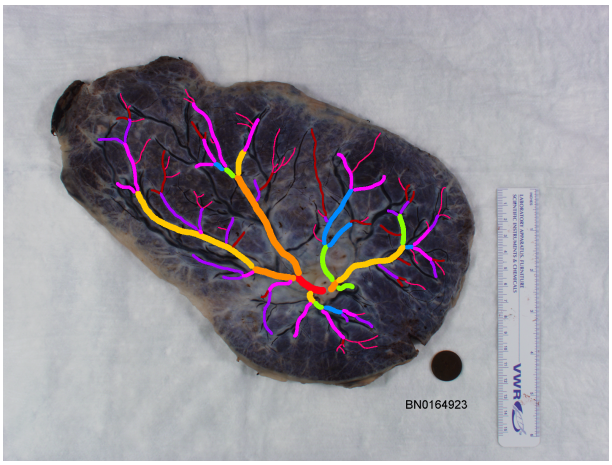
(a) Fixed Placental Sample



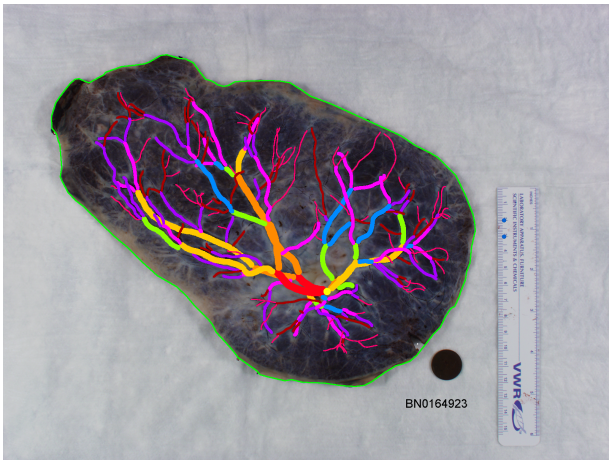
(b) Perimeter tracing and Umbilical cord insertion point (UCIP



(c) Arterial tracing



(d) Venous tracing



(e) Total Vascular Network

FIGURE 11: A representative placental sample and tracing

vessel width	color (hex value)	color name
3 pixels	#ff006f	magenta
5 pixels	#a80000	dark red
7 pixels	#a800ff	purple
9 pixel s	#ff00ff	light pink
11 pixels	#008aff	blue
13 pixels	#8aff00	green
15 pixels	#ffc800	gold
17 pixels	#ff8a00	orange
19 pixels	#ff0015	bright red

TABLE 6: Vessel width color code for manual tracing protocol

diameter are either binned to three or (quite frequently) left untraced. The correspondence between pencil color and (binned) vessel width used in the tracing protocol is given in Table 6.

As stated in the introduction, the task of creating these samples, in particular the tracing and estimation involved in creating Fig. 11c and Fig. 11d is very labor intensive–requiring between 4 and 8 hours to trace a single sample. A closer look at many of the samples often reveals that a great deal of subjectivity in providing this "ground truth," as it is not often clear what the underlying truth really is. Often it's hard to see where the vein is, vascular networks are obscured by the umbilical stem, the blood in the vessels dries unevenly or ruptures, and the vessel seems to disappear momentarily. These situations and more will be showcased in Fig. 18, where we will demonstrate how the Frangi filter reacts to these problem areas. Our efforts at the eventual task of network completion must deal with these shortcomings and, in some circumstances, make subjective decisions like the manual tracer did.

7.1.2. Knowns and Unknowns

Since our final goal is a fully automated procedure, we wish to simply operate on the placental sample itself, without any understanding of its provided tracing (except for judging the strength of our algorithm); our goal is to develop an algorithm that can produce a "ground truth" tracing similar to Fig. 11e or Fig. 12d without any user intervention.

For our purposes however, we will concede and provide a limited amount of information

from the tracings, namely the provided placental perimeter (shown in green in Fig. 11). In developing a fully automated algorithm, it would be relatively straightforward to obtain this boundary ourselves using various techniques, such as an Active Contour Model [25] or, even a simple edge finding algorithm followed by watershedding and largest object selection. In fact, we have implemented this later algorithm, but unfortunately we achieve a subpar result on several images, and therefore we currently use it to improve upon the perimeter by removing "cuts", which were previously reported as being inside the plate and sometimes led to large false positives.

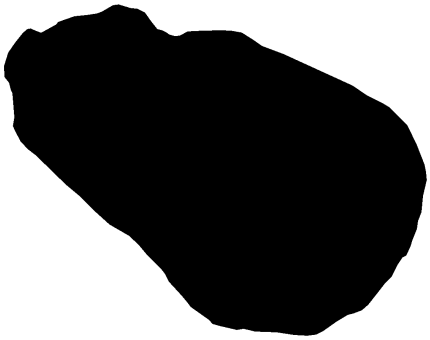
Finally, we will consider the location of the umbilical insertion point as a "known", as the vessels around it are frequently impossible to see and we wish to exclude them from consideration. It is not unreasonable, however, to consider this to be a known–in future preparations of samples, we could simply require that this point be centered in image in a predictable location. Furthermore, we use its location as a convenience in data analysis–knowledge of this point does not inform our algorithm at the present time.

7.2. Data Cleaning and Preprocessing

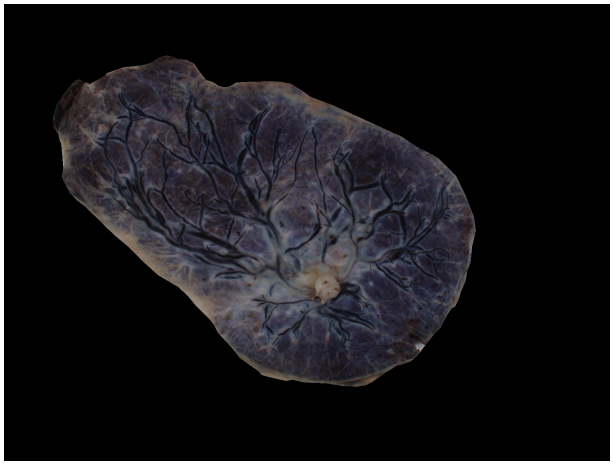
Building a sample suitable for use in our algorithm from Fig. 11 is relatively simple. We zero outside the boundary of the plate (so as to not waste computational time calculating the differential geometry of a ruler, say), and also generate a binary mask to identify the plate. Finally, our vessel layers are combined and given as a binary trace, which we will use later for scoring. An example of the preprocessed samples used by the algorithm are given in Section 7.2.

These procedures are performed automatically on the 201 images in our data set using a custom GIMP plug-in, which performs various "bucket fill" operations, layer mergings, and thresholdings. For completeness sake, this plug-in (and an associated Scheme script which turns it into a batch operation) can be found in the Appendix.

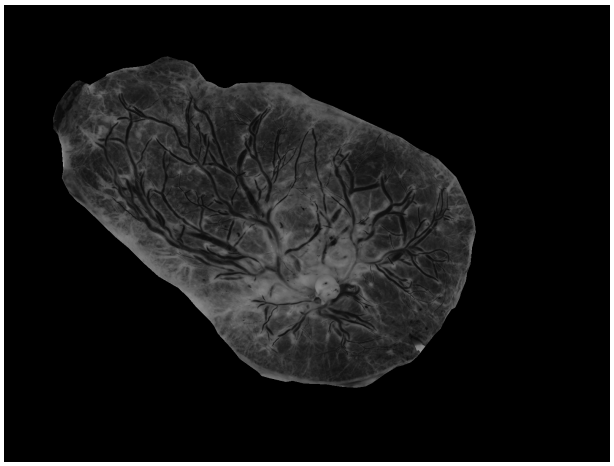
As a point of technicality, the grayscale image in Fig. 12c is not actually produced directly by the extractor plug-in, but created when the 3 channel RGB image Fig. 12b is imported at the start of the algorithm. This grayscale conversion is simply done for ease of analysis on the sample:



(a) Background mask (in white)



(b) Sample with background removed



(c) Grayscale

(d) Trace / "Ground Truth"

FIGURE 12: Preprocessed files from an NCS sample

although the Frangi filter is designed for arbitrary N-dimensional input [12], an image with three color channels does not have 3 spatial dimensions. We therefore simply combine the information in three channels using the well-known and oft-implemented ITU-R 601-2 luma [26], or "luminance" transform:

$$L = \frac{299}{1000} R + \frac{587}{1000} G + \frac{114}{1000} B \tag{7.1}$$

It should be noted that this choice is not a given–several other attempts have used the green channel unmodified, as in [5] and [2]. Preliminary and periodical rechecking has not indicated that such a conversion has any benefit over the luminance transformation for our image domain, although other placental samples with a different preparation method might benefit from it.

7.2.1. Boundary Dilation

All images are grayscale, M, N pixels as a masked array (of type `numpy.ma.MaskedArray`), where pixels outside of the placental region are masked so they will not be considered by the algorithm. However, some standard implementations of algorithms, namely `numpy.gradient and scipy.signal.convolve2d` are not designed to handle masked regions. Although it would be potentially useful to adapt such methods in a way to, say, calculate a gradient or perform a convolution by a "reflection" across an arbitrary closed boundary (as opposed to the edge of the image matrix), we opted instead to "zero out" unwanted background pixels and simply exclude affected areas from consideration. This exclusion could be achieved by simply dilating the mask, but we opt to achieve it in a much more resource efficient manner: we iterate through an array of indices for the image where the boundary occurs and simply extend the mask R pixels in each direction (like a giant plus sign). Since the boundary of the placental plate forms a closed loop, the effect is very similar to convolving with a disk of radius R, but is much faster.

Fig. 13 shows the effect of this so-called "boundary dilation." We show the output of the Frangi filter with $\sigma = 3$. A border radius of 25 is chosen to exaggerate the effect. The first row

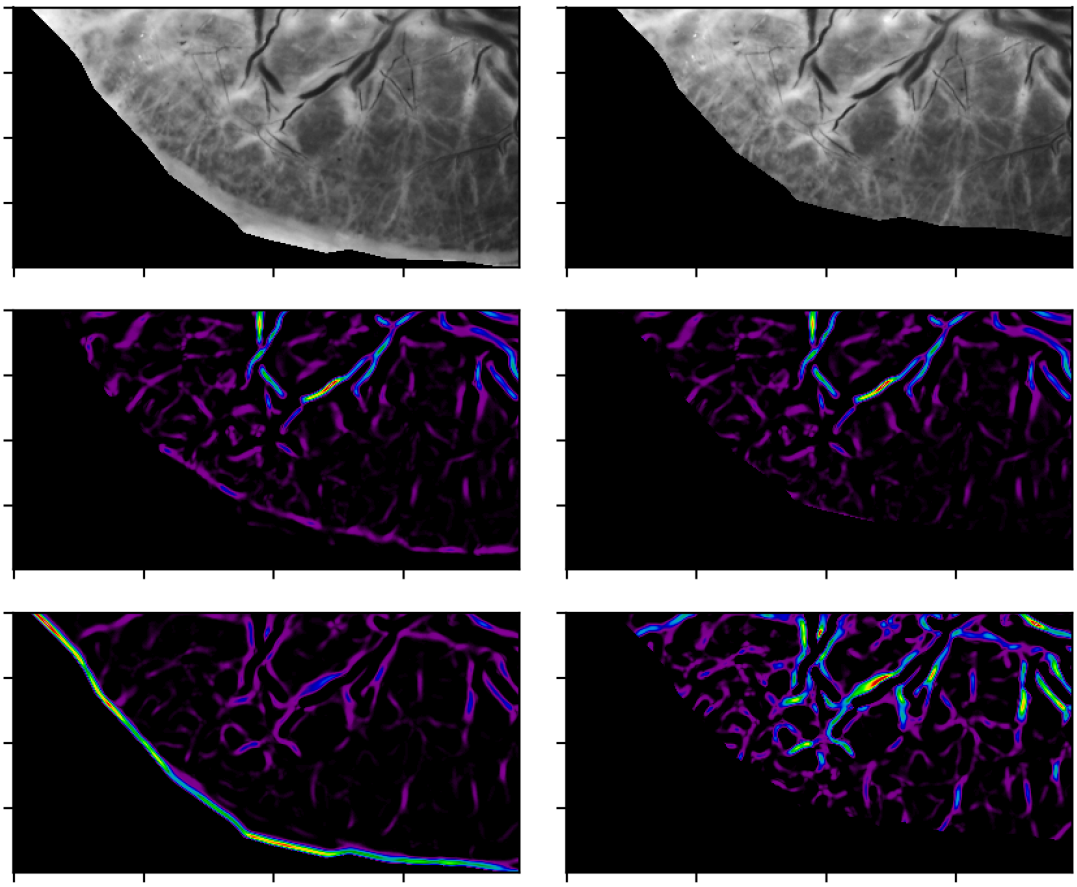


FIGURE 13: Effect of boundary dilation on Frangi responses

shows the unaltered boundary of the sample (left) and the sample after boundary dilation (with radius dilation of 25 pixels). The second row shows the Frangi vesselness measure at single scale ($\sigma = 3$) where `DARK_BG=False` to target dark curvilinear structures performed on the altered sample (left) and the boundary dilated sample (right). Removing an unnecessary part of the placental plate prevents a small response to a non-vascular yet mildly curvilinear background feature from appearing. The third row of Fig. 13 shows the Frangi vesselness measure at the same scale ($\sigma = 3$) when we are probing for bright curvilinear structures (i.e. `DARK_BG=True`). Here, wherever the very edge of the placental plate is *any* brighter than adjacent interior, a very large Frangi response will occur, as seen on the left. Dilating the boundary completely avoids this issue, as seen by the figure on the right. Thus we prevent a visual artifact that is present in much prior work on this problem (see [2], [5]). It should be noted that, while the figure on the right shows a much larger interior response, this is simply because the intensity of the output in each of these images is being independently scaled between the minimum and maximum intensity in the image. However, we argue that this is an appropriate and desired depiction of the situation, as we will frequently consider only the relative maxima of Frangi response per scale in our analysis.

We end our discussion by noting that we perform this boundary dilation within the Frangi algorithm itself when we set the structureness parameter γ as half of the maximum Hessian norm found at that scale–this ensures that the maximum occurs sufficiently away from the boundary of the plate, and does not occur from a noise phenomenon.

The code for generating Fig. 13 is found within the "`if __name__ == __main__`" block of the file `plate_morphology.py`, (so the figure will be generated when running `plate_morphology.py` as a top-level script from the command line). See appendix.

7.2.2. Finding Placental Border

We take the gradient of the sample green channel of the raw sample with a very small scale $\sigma = .01$ to identify very small and sharp changes. We then mark the top right pixel, which is certainly outside the placental plate, with a background marker, and we label any points where the gradient is above its mean value as foreground. We then perform a watershed of the gradient given these markers and then perform a binary erosion of the result with radius 15 to regularize the result. From this, we grab the largest foreground object, which will inevitably be the placental plate. Unfortunately, there are 7 of the 201 samples for which this procedure fails. In the case of these samples, we revert to simply using the traced outline of the placenta as given by the tracing protocol.

7.2.3. Deglaring

Despite best efforts when harvesting samples, a select number of the placental samples exhibit substantial glare, which leads to inaccuracies in identifying curvilinear content. Our protocol for deglaring is analogous to that performed in [5] and [2]. Unfortunately, the method relied upon by those previous papers (`MATLAB`'s `imfill`, which relies on inpainting by solving the Dirichlet problem for masked regions) was not immediately available in a Python environment. Instead, we used an already implemented inpainting algorithm, `scikit-image`'s `inpaint_biharmonic()`, which should be expected to achieve similar results, albeit at the expense of processing time.

The function `inpaint_biharmonic` is based on [27], and relies on solving the biharmonic equation i.e. $\nabla\nabla f = 0$ for the surface f subject to boundary conditions (as compared to `imfill`'s solving the Laplace equation $\nabla f = 0$ in regions marked as glare.

The method for deciding what is considered glare is similar to [5], in which we consider any intensities close the maximum intensity in the image (Almoussa et al. used 80% of max intensity, and we use $175/255 \approx 68\%$). This threshold is unfortunately dependent on the image domain.

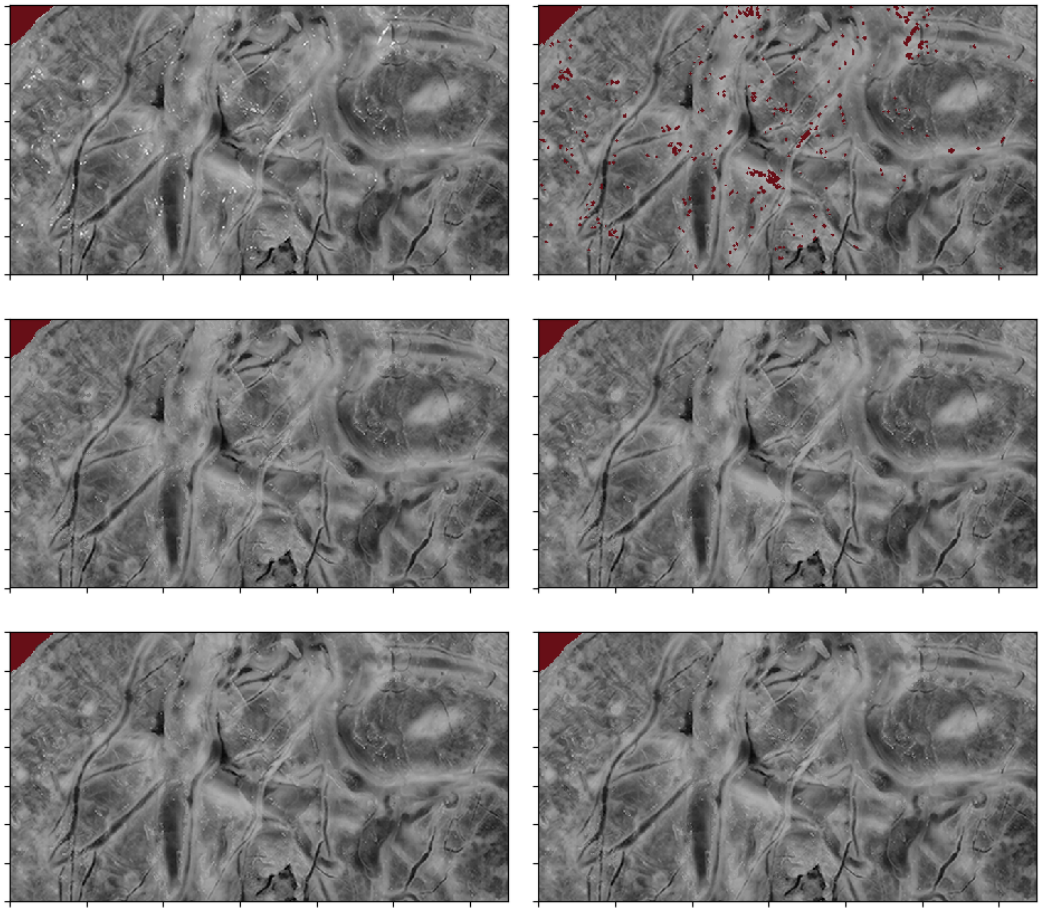


FIGURE 14: Deglaring a sample using a hybrid inpainting method

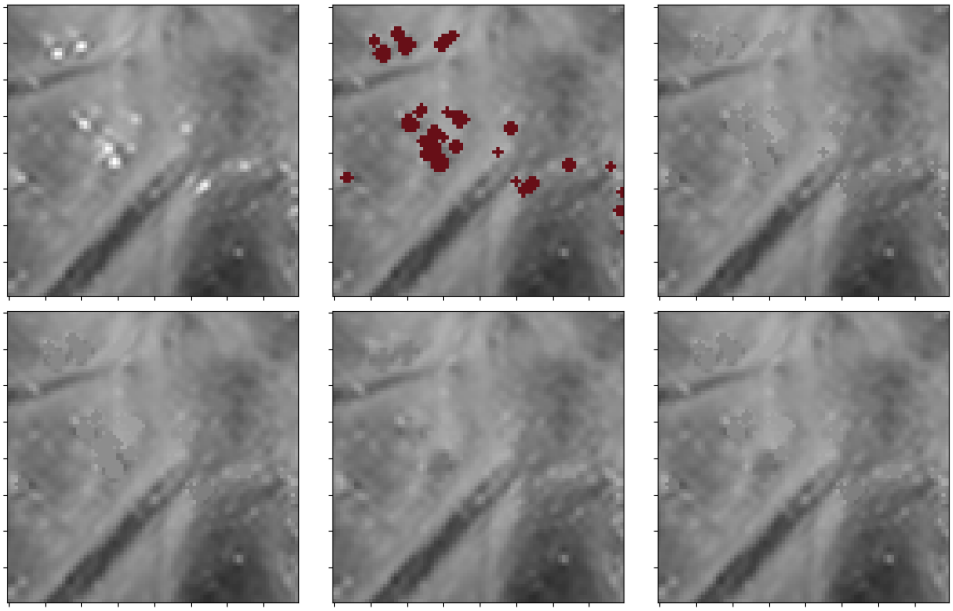


FIGURE 15: Comparison of glare inpainting methods (detail)

Inpainting in the above way is rather resource intensive, so we implemented two faster and less precise methods of inpainting that work well enough for removing small regions of glare. These methods can be found in `preprocessing.py`. The first, called `inpaint_glare()`, replaces any masked pixel with the average of all non-masked values within a certain distance (default 15 pixels). The second, called `inpaint_with_boundary_median` calculates the median value of the (non-masked) boundary and fills any masked region with that value. We argue that these less-exact methods are adequate for smaller regions, while larger regions of glare deserve a more thoughtful application of inpainting. Our final method of inpainting, `inpaint_hybrid` implements this idea–smaller glare regions are inpainted with a boundary median, while larger areas are inpainted with the more expensive but more accurate biharmonic inpainting.

A comparison of these methods is shown in Fig. 14, and a zoomed in portion is shown in Fig. 15. In the top left of each, the glary image is shown. In the top middle, regions above the threshold intensity are masked (shown in dark red, along with the background). In the top right,

the strategy is "mean window" with a window size of 15 pixels. The bottom left uses "boundary median" strategy. The middle is the more expensive "biharmonic inpainting" strategy, and the bottom right uses a "hybrid" strategy.

The following timing demonstrates that the "hybrid" strategy is over 3 times faster than biharmonic inpainting, and that biharmonic painting takes 22 seconds, even when only 1% of the placental plate is to be inpainted.

Inpainting with boundary median only four seconds, inpainting all sections with the biharmonic method took 22 seconds, and a hybrid method took 6.5 seconds. Roughly 1 percent of nonmasked pixels in the image were inpainted.

We stress again that only a small subset our image domain exhibits disruptive amounts of glare for anything except small scales. Future improvements in this direction should probably seek to implement more robust method such as [28] that are not dependent on an arbitrary global threshold for deciding what regions exhibit glare.

7.3. Multiscale Setup

Our multiscale Frangi filter requires a list of scales at which to probe. Each scale is chosen to accentuate features (i.e. vessel diameter) of a particular size. This set of scales at which to probe will be denoted as $\Sigma := \{\sigma_1, \sigma_2, \ldots, \sigma_N\}$, where each scale in indexed by increasing order.

Although we cannot expect *a priori* that there is an direct proportionality between our scale size σ and (even some function of) the width of a particular vessel [12], we generally expect to isolate narrower curvilinear structures at smaller scales, and thicker curvilinear structures at larger scales. The smallest one should be an effective size where details are expected to be isolated, and the largest should be an effective size as well. In fact, following [16], it is reasonable and natural to select these logarithmically; that is, for some selected inputs $m < M$ we have

$$\sigma_1 = 2^m \, , \, \cdots \, , \, \sigma_j = 2^{\left(m + \frac{M-m}{N-1} j\right)} \, , \, \cdots \, \sigma_N = 2^M \tag{7.2}$$

That is, the exponents are spaced linearly from m to M. This is achieved by the command

`np.logspace(m,M,num=N)`. The idea is that the curvilinear content of the image will respond better at some particular scale, but there are diminishing returns as σ increases; while the filter's response may vary substantially between, say $\sigma = 1$ and $\sigma = 2$, we would not expect a substantial difference in response between, say, $\sigma = 46$ and $\sigma = 47$. Historically, there was another benefit of using a logarithmic scale space: computing the vesselness measure was very expensive, and thus it was simply not feasible to collect so many large scale readings. This is much less of an issue with the present implementation, although we still obviously wish to avoid frivolous, redundant calculations no matter the speed of the implementation.

The optimal choice of scale sizes to probe at is intuitively dependent on the resolution of the image. If there is no particular care taken in selecting a minimum and maximum scale at which to probe, then we must assure that our Frangi filter is "normalized" in such a way that there is a decay in response past certain values. That is, probing at a unreasonably large (or small) scale (say $\sigma = 1000$ or $\sigma = .0001$ should result in an almost null response throughout the image. We will approach this issue in our discussion of "variable thresholding."

7.4. The Research Protocol

Once we have chosen this set of scales Σ, we simply convolve the image with a discrete Gaussian kernel with that standard deviation, then take gradients enough to get a matrix of partial second derivatives, the Hessian. We calculate the eigenvalues of each (2x2) Hessian matrix and then compute the Frangi filter according to Chapter 3 and Chapter 5. We use these to provide a couple examples of estimating the PCSVN network. The entire decision tree can be shown in the outline below. We follow the procedure for Chapter 9. For our analysis of the Frangi filter apart from segmentation we shall stop before step (C).

```
 % DECISION TREE
For each sample:

A) Preprocessing
  1) Find placental border
   a) Gradient of green channel for small sigma=.01
   b) Watershed outside against above gradient mean
```

```
   c) erode these points, radius=15
   d) Identify largest object in watershed threshold
  2) RGB to single channel via Luminance Transform
  3) Glare removal
    a) Mask glare (threshold at *175/255*)
    b) Dilate glare mask with radius=2
    c) Inpaint glare
      + Hybrid inpainting, with size threshold *32*
      . Biharmonic inpainting
      . Median value of boundary
  4) Dilate around UCIP, add to mask (radius=50)
B) Multiscale Frangi filter
  1) Set parameters
    a) Scales
      = n_scales (default: *20*)
      = log_range (default *[-1.5, 3.2]*), log base 2
      -> scales
    b) Beta = 0.1, Gamma = 1.0 (or alternate parametrization)
    d) Dilate mask per scale (20 pixels)
  2) For each sigma: compute Uniscale Frangi Filter
    a) gauss blur image with discrete Gaussian kernel FFT
    b) take gradient across each axis of blurred image;
       take gradient across each axis of gradient
       -> (Hxx, Hxy, Hyy)
    c) Find eigenvalues of hessian at each point (using np.eig)
       and sort by absolute magnitude
    d) Calculate Frangi Vesselness Measure
  3) Split positive and negative strain.
  3) Merge each result, reserve positive and negative stacks
C) Estimate PCSVN
  1) Approximate using strategy
    a) Calculate Vmax above high fixed threshold
    b) Calculate Vmax above lower fixed threshold
    b) Threshold scalewise at 95th nonzero-percentile
    c) Threshold scalewise at 98th nonzero-percentile
    d) Margin adding algorithm with high fixed threshold
  2) Compare to Trace to obtain confusion matrix
  3) Calculate MCC score and precision
```

CHAPTER 8

RESULTS AND ANALYSIS

We demonstrate the output of the Frangi filter on our samples after running a multiscale technique with $N = 20$ scales with a stricter anisotropy parameter $\beta = 0.35$ and standard structureness parameter $\gamma = 0.5$, with scales spaced logarithmically from $\sigma_1 = 2^{-1}$ to $\sigma_N = 2^{3.5}$, performing glare and cut removal in preprocessing, and using a discrete gaussian kernel and dilation border of 20. Our goal in this section is to provide a close up look at the Frangi filter on two samples, and then provide some measures of the Frangi output's correspondence with the "ground truth" network tracings across all 201 images, without explicitly performing segmentation. However, for visual demonstration, we will employ both simple thresholding techniques (arbitrary fixed and nonzero-percentile) described in Chapter 5. In Chapter 9 we will develop and analyze more sophisticated Frangi-based segmentation techniques and compare their performance to these rudimentary thresholding techniques across the entire dataset. For this chapter we will content ourselves with analysis of the raw Frangi vesselness measure within our image domain, and use our simple thresholded results for visual demonstration alone.

8.1. Sample Visual Output

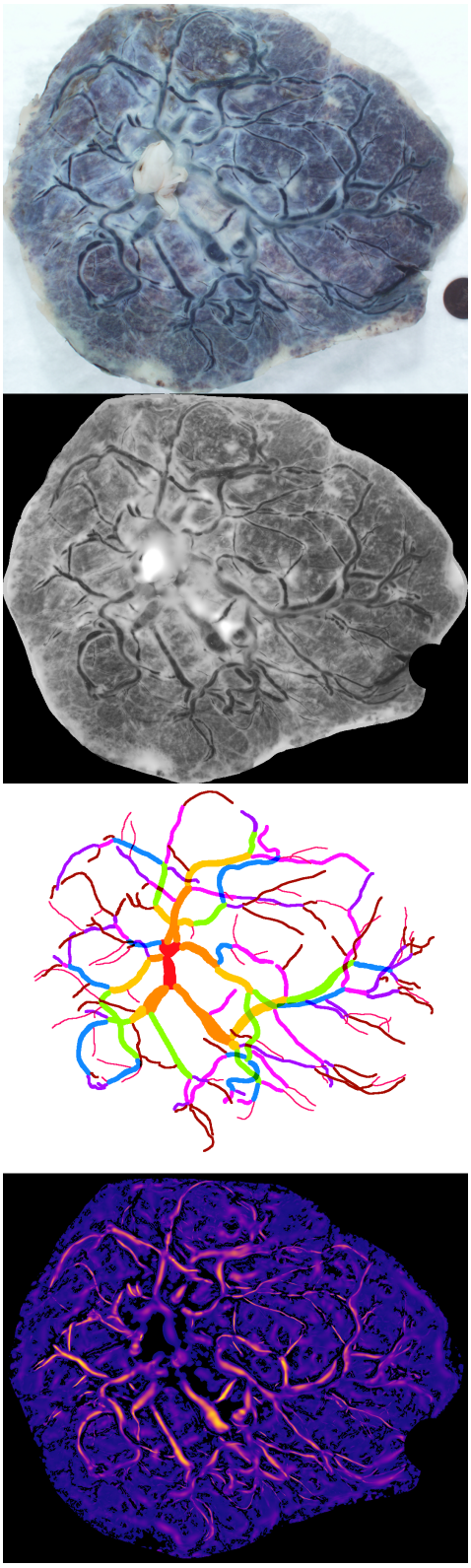
In Fig. 16 and Fig. 17 we take a look at the Frangi output ($\mathcal{V}_{\max}$) for a well-behaved sample. In the left column we have (from top to bottom) the un-preprocessed placental sample, the preprocessed sample, the color trace (ground truth), and $\mathcal{V}_{\max}$. In the top-right, we have for each $n = 0, \ldots, 19$ scales, we have the size of the scale σ_n, the 95th percentile score at that scale, i.e. α_p where $p = 95$, and the maximum vesselness score given to a pixel at that scale, $\max(\mathcal{V}_\sigma)$. This table is mostly to show that $\mathcal{V}_\sigma$ does decay slowly as σ increases, and that smaller scales experience slightly dampened fbut This is The bottom-right set of images are two rudimentary

thresholds of the Frangi result. The top is the scalewise nonzero percentile threshold (95th percentile) and the bottom is a strict threshold of $\mathcal{V}_{\max} \geq 0.4$. For each pixel that passes the threshold, each of these graphs is color coded to indicate from which scale the maximum value at that pixel occurs, colored according to the scale on the left. These two thresholds are included mostly to further show the dependence of scale on which vessels are identified.

8.2. Ground Truth Imperfections

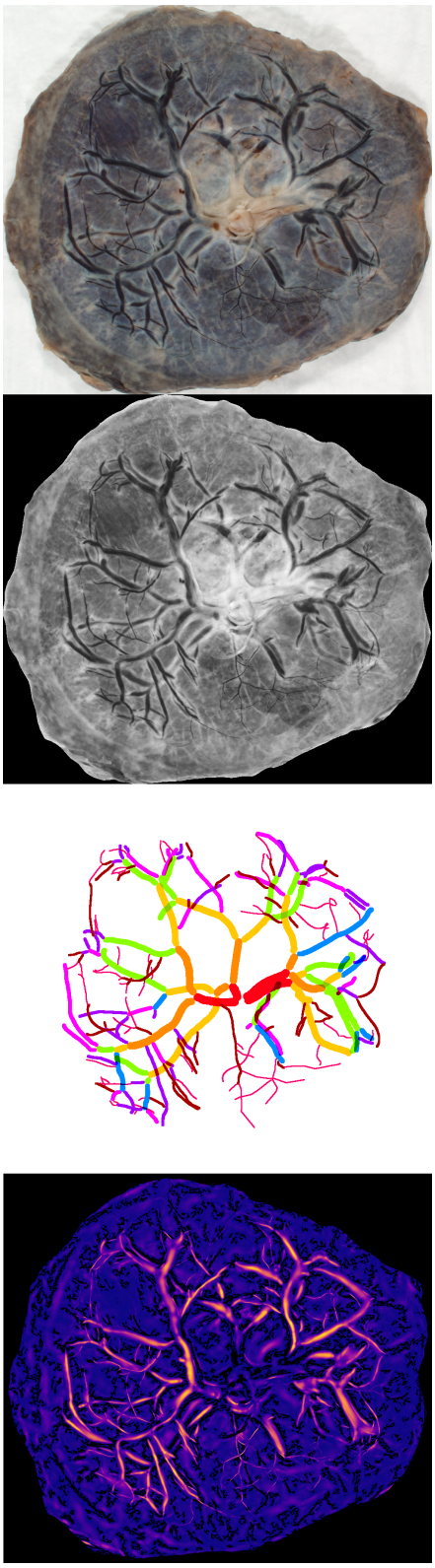
We must first qualify binary classification. There are limitations intrinsic to the image domain and tracing protocol that make our ground truth tracing somewhat inaccurate. In Fig. 18 we demonstrate a few common issues with the samples. The four figures show (top left) the original colored raw sample, (top right) the ground truth tracing, (bottom left) $\mathcal{V}_{\max}$, and (bottom right) the confusion matrix after some segmentation strategy. The green arrow points to the umbilical stump. You can see there is circular noise around this point, and in general perfusion around this point is very low–many of these vessels have been estimated by the tracer. The orange arrows represent points where perfusion is very low, perhaps caused by a clot in the blood vessel. The pink arrows point to vessels that were not traced, although they are clearly visible. The blue arrows point to where the shape of the vessel and the trace clearly do not agree. There any many more examples of these in the frame, and many more across all samples. The red arrow points to an issue unrelated to the ground truth, but an issue that arise when selecting scales in our multiscale methodology– this is a point of dark curvature that represents noise at larger scales. As you can see from the $\mathcal{V}_{\max}$ of this inset, there is a positive response in this dead space between vessels, and there is another one that appears in the bottom left between two close vessels of similar size.

The sample we examined was a relatively well-behaved sample compared to others. Conversely, in Fig. 19, we give examples of some sampled that did not respond well to the Frangi filter, nor any of our Frangi-based segmentation methods. As can be seen from this gallery, poor perfusion of the blood vessels and large amounts of high constrast background is the common



n	σ_n	α_p	max(V_σ)
0	0.353	0.054	0.986
1	0.424	0.059	0.979
2	0.509	0.065	0.970
3	0.611	0.076	0.973
4	0.733	0.089	0.988
5	0.880	0.096	0.991
6	1.056	0.108	0.991
7	1.267	0.130	0.970
8	1.521	0.166	0.973
9	1.825	0.223	0.978
10	2.190	0.292	0.984
11	2.629	0.319	0.968
12	3.155	0.326	0.994
13	3.786	0.355	0.998
14	4.544	0.405	0.999
15	5.454	0.376	0.963
16	6.545	0.318	0.950
17	7.855	0.304	0.958
18	9.427	0.328	0.916
19	11.313	0.352	0.916

FIGURE 16: Vesselness score, percentile thresholds, and simple thresholds, Example 1



n	σ_n	α_p	$\max(V_\sigma)$
0	0.353	0.054	0.999
1	0.424	0.059	0.999
2	0.509	0.065	0.999
3	0.611	0.076	0.999
4	0.733	0.089	0.999
5	0.880	0.096	0.981
6	1.056	0.108	0.916
7	1.267	0.130	0.889
8	1.521	0.166	0.917
9	1.825	0.223	0.938
10	2.190	0.292	0.935
11	2.629	0.319	0.961
12	3.155	0.326	0.987
13	3.786	0.355	0.987
14	4.544	0.405	0.971
15	5.454	0.376	0.991
16	6.545	0.318	0.888
17	7.855	0.304	0.876
18	9.427	0.328	0.930
19	11.313	0.352	0.900

FIGURE 17: Vesselness score, percentile thresholds, and simple thresholds, Example 2

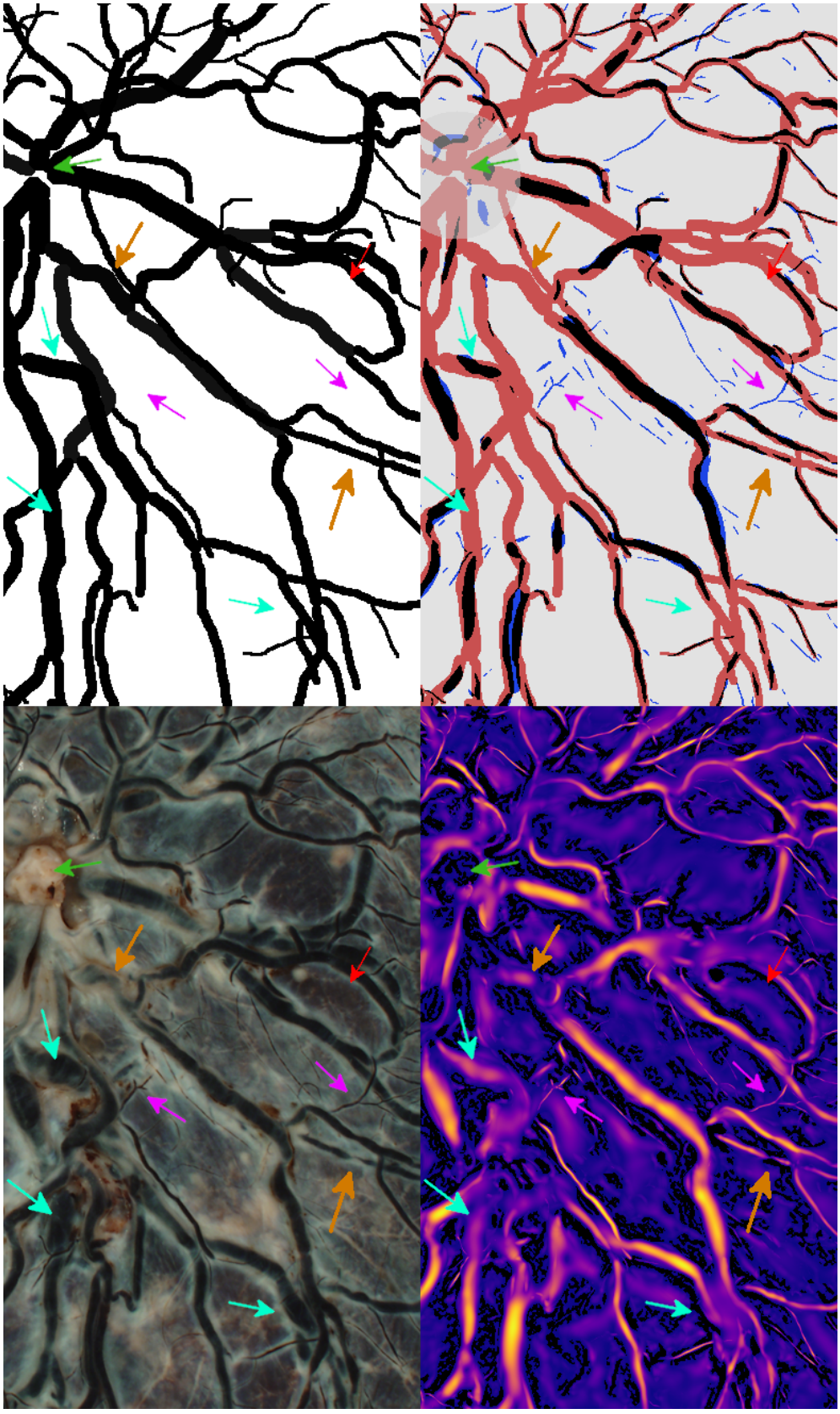


FIGURE 18: Issues with the ground truth manifesting in Frangi vesselness scores

thread. One particular sample has very large arteries that were not reported. Increasing the scale size might fix the issue for this particular sample.

We have empirically identified a few reasons for issues. Although we have in many cases removed border areas from consideration where they would cause a large false response (see Section 7.2.1 for our discussion on boundary dilation), some samples still demonstrate noise around the "collar" of each plate. This can marginally be seen in Fig. 17–there is a large Frangi response in the bottom left that was not removed, and thus shows up in each simple segmentation. In Chapter 9 we have opted to remove an area around the umbilical cord insertion point from our Frangi calculation for the same reason, although in Fig. 18 it is unmasked its effect on the Frangi output.

8.3. Results

In Fig. 20, we demonstrate the usefulness of scricter parameters for the Frangi filter. Since the Frangi vesselness measure is a "probability-like" score, we should be interested–without doing any actual segmentation yet–to what extent this score aligns with the ground truth in a cumulative sense. We should hope at least that larger values of our vesselness measure should occur along vessels and not within the background. We define the cumulative vesselness ratio by "integrating" the max vesselness score over pixels in the ground truth and over the entire image and considering the ratio:

Definition 8.1. *The cumulative vesselness ratio for a particular parametrization of the multiscale Frangi filter is given by*

$$CVR(\mathcal{V}_{\max}) := \frac{\sum_{G \subset \mathtt{I}} \mathcal{V}_{\max}(x_0, y_0)}{\sum_{\mathtt{I}} \mathcal{V}_{\max}(x_0, y_0)} \tag{8.1}$$

where the sums on top and bottom are being carried out for all pixels (x_0, y_0) *in the "ground truth" subset* G *in the image* `I` *and over the entire image* `I` *itself, respectively.*

Fig. 20 and Fig. 21 show $\mathcal{V}_{\max}$ for two well-behaved samples with reported CVR for 9 various combinations of β and γ. Two combinations in particular, labeled "Anisotropy Factor"

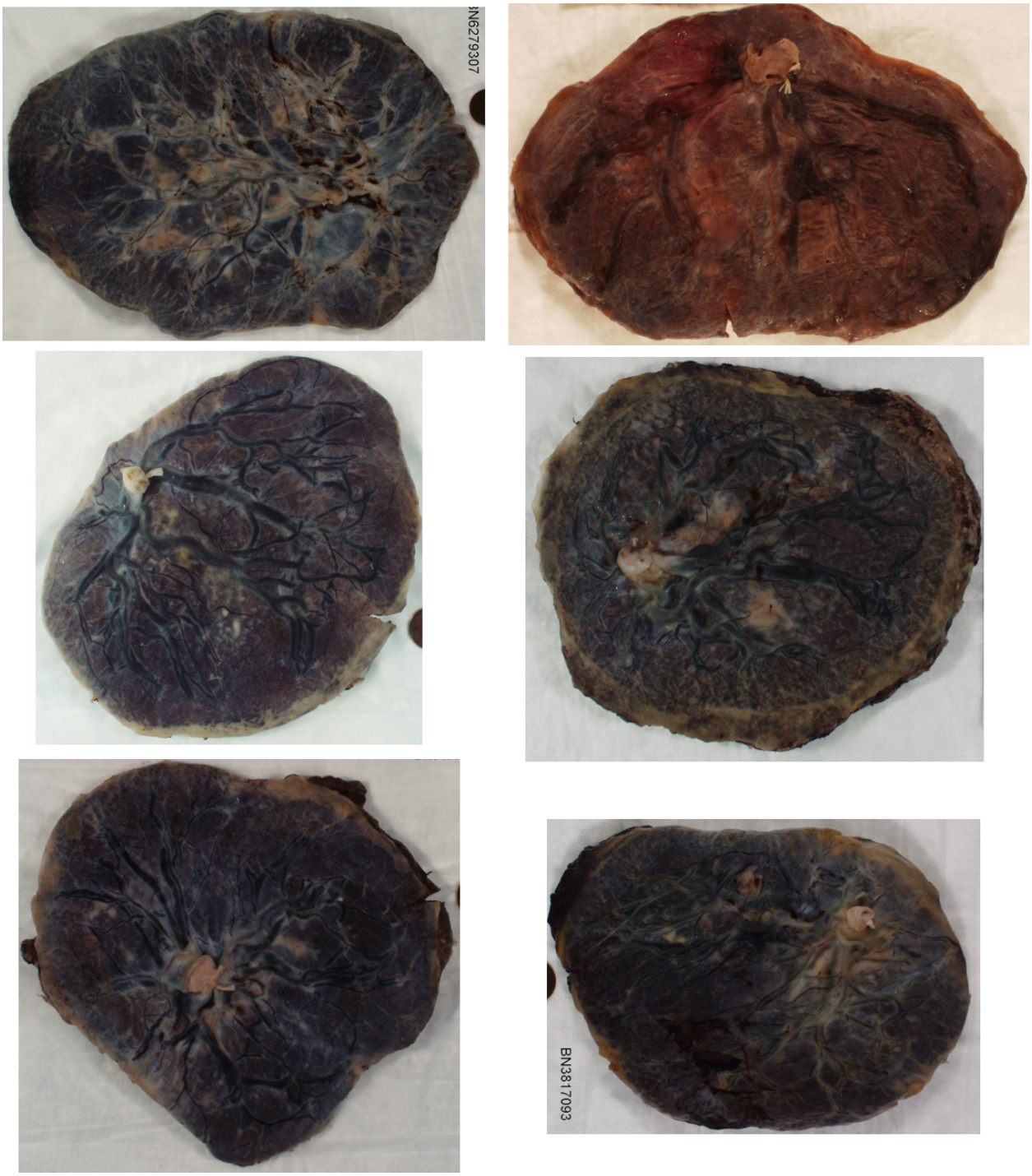


FIGURE 19: Bad samples

and "Structureness Factor," have parameter choices so that the structureness and anisotropy factors (respectively) are exactly one throughout the entire image. Comparing all parameter combinations, we see in both examples that stricter parameters (smaller β and larger γ) correspond to an increased CVR. That is, the more selective our Frangi filter is, the better it aligns with the vascular content within the image. This suggests not only that the Frangi filter is well suited to this image domain, but also that choosing stricter parameters for the Frangi filter concentrates the Frangi response within vascular features more and reduces background noise. We also notice that maximum structureness factor throughout the entire image behaves relatively similarly to the Frangi filter itself at this scale range, although much more of the background is filtered out by incorportating the anisotropy factor.

Each of these sample was run over the same scales (logarithmically spaced from $2^{-1.5}$ to 2^3 with $N = 12$ and the color output of each $\mathcal{V}_{\max}$ is independently normalized between 0 and 1.

In Fig. 22 we show how the CVR is affected for 25 of the "best" samples, that is, those that performed better in segmentation techniques relative to other samples, regardless of Frangi parametrization or particular segmentation method. This shows that the relationship between parametrizations of our two examples seem to hold in general for all "well-behaved" samples in our image domain.

The upper image in Fig. 23 depicts, for a single sample, the relationship between the scale at which $\mathcal{V}_{\max}$ (which we will call $\mathcal{V}_{\text{argmax}}$) occurs for a particular pixel and the width of the vessel according the ground truth, and the bottom figure is a coloration of the ground truth with this same information. We can see that in general, larger scales correspond with larger reported vessel widths, and smaller scales correspond with smaller reported vessel widths. In Fig. 23 we repeat these same observations but limit ourselves only to those pixels identified in the ground truth for which some $\mathcal{V}_\sigma$ occured above some scale's 95th percentile (as will be explained in Chapter 9, these pixels are the "true positives" of the nz-percentile segmentation method). The y-axis of each of the charts is the ratio of pixels at which $\mathcal{V}_{\max}$ is attained at the given width over all scales for

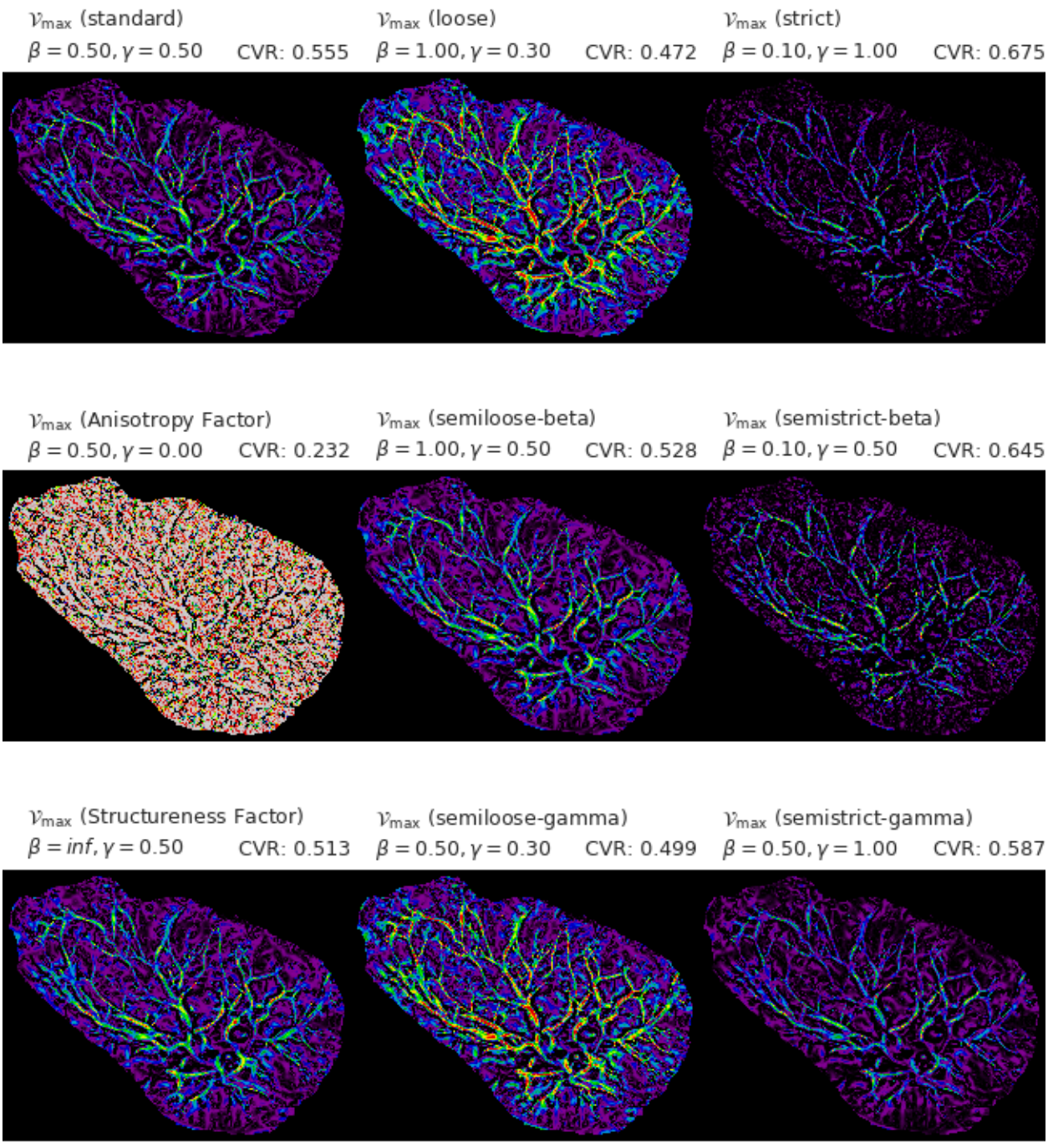


FIGURE 20: $\mathcal{V}_{\max}$ and CVR for varying multiscale Frangi parametrizations (Example 1)

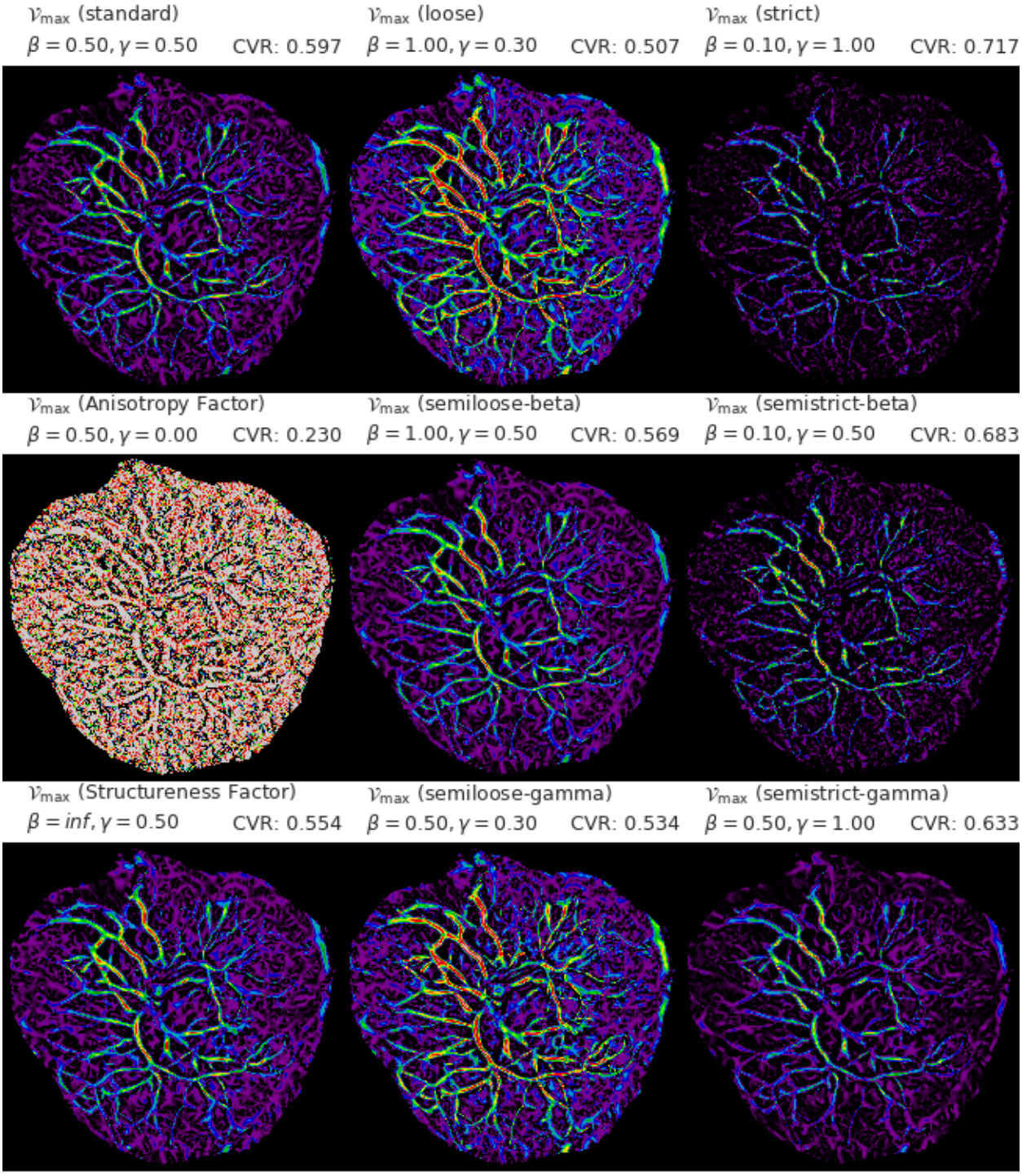


FIGURE 21: $\mathcal{V}_{\max}$ and CVR for varying multiscale Frangi parametrizations (Example 2)

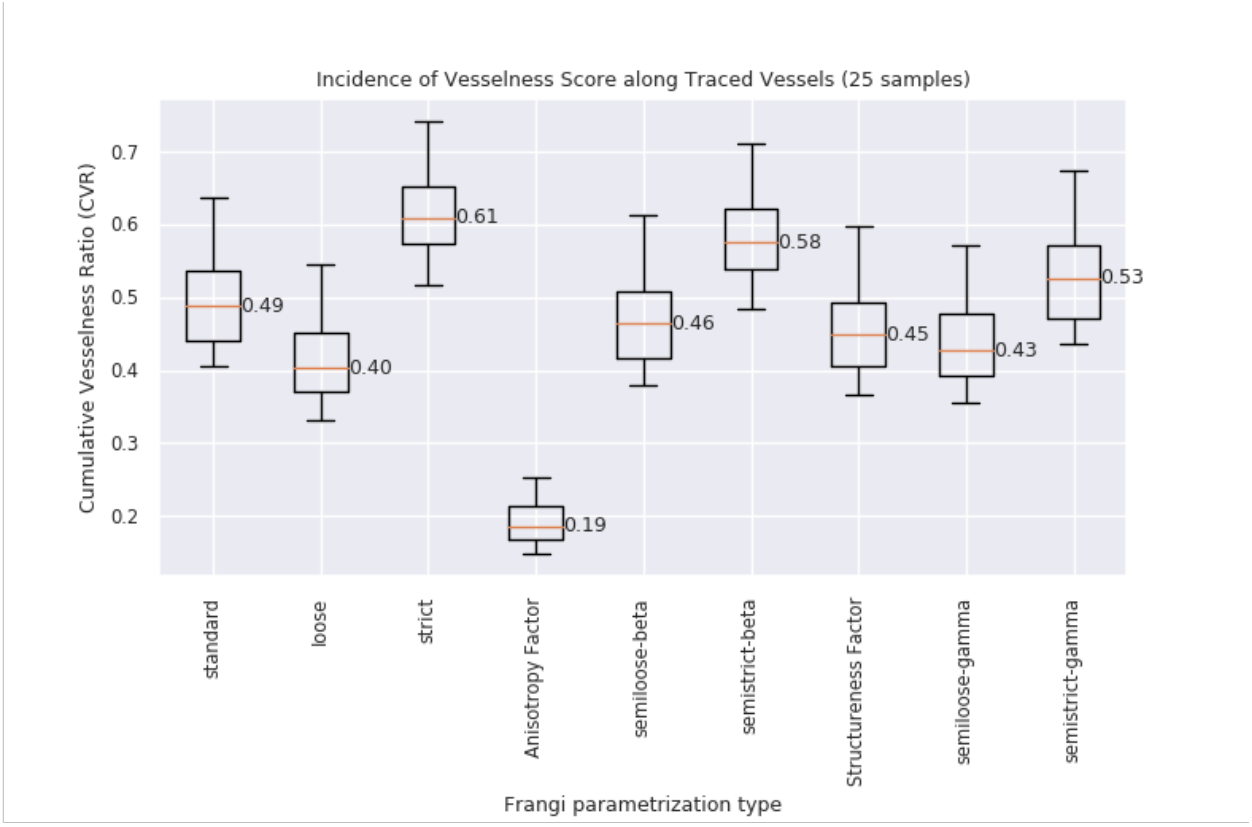
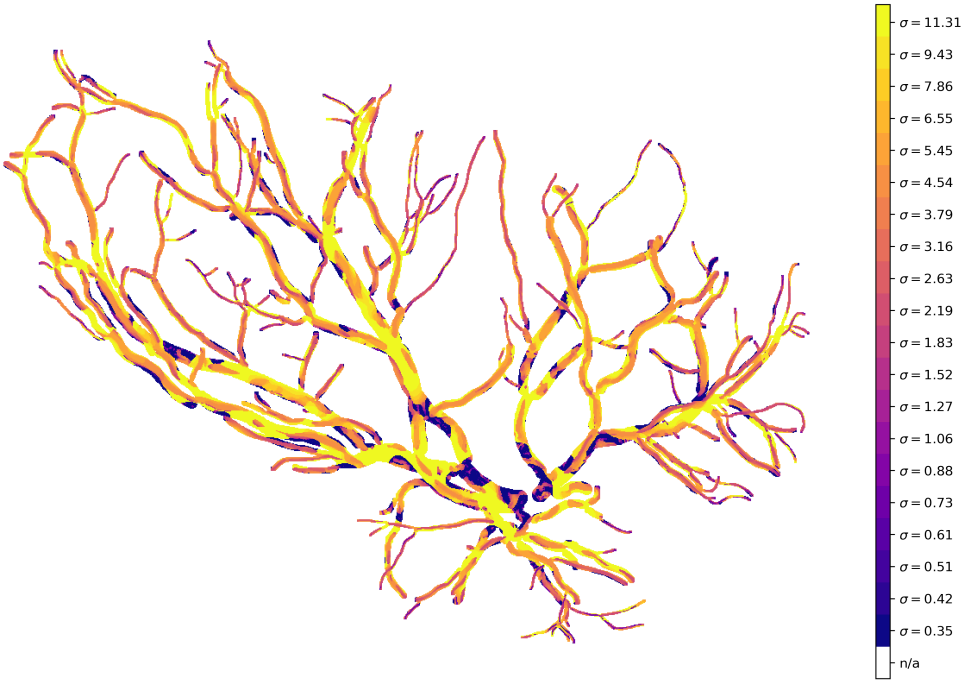


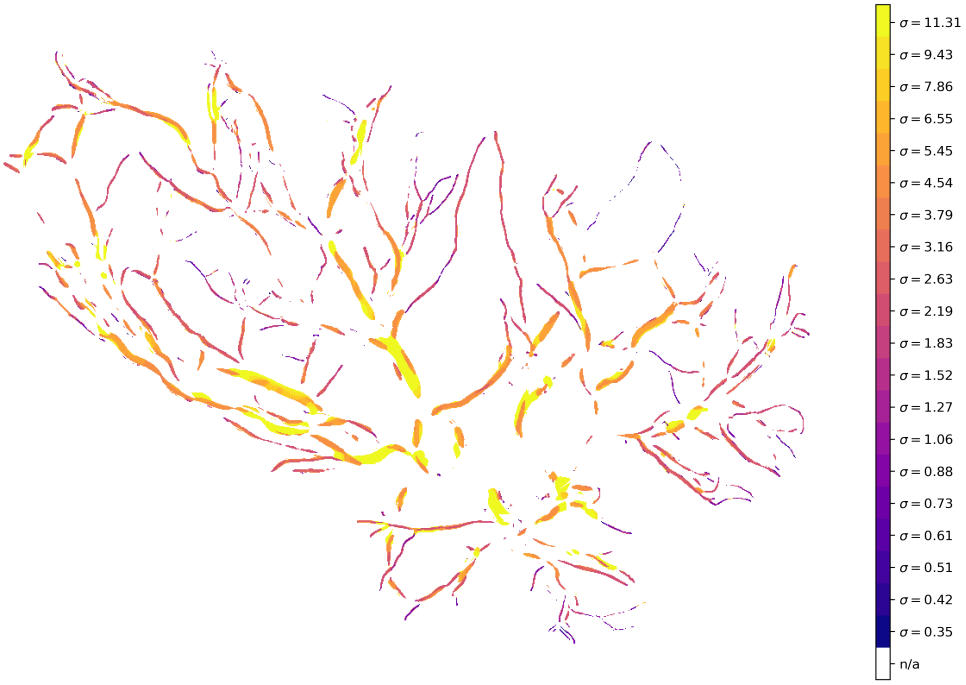
FIGURE 22: *CVR* scores of 25 samples under varying parametrizations

Label	β	γ
standard	0.5	0.5
loose	1.0	0.3
strict	0.10	1.0
anisotropy factor	0.5	0
semiloose-beta	1.0	0.5
semistrict-beta	0.1	0.5
structureness factor	∞	0.5
semiloose-gamma	0.5	0.3
semistrict-gamma	0.5	1.0

TABLE 7: Summary of Frangi parametrizations for CVR demo



(a) $\mathcal{V}_{\text{argmax}}$ along ground truth



(b) $\mathcal{V}_{\text{argmax}}$ along ground truth (true positives of segmented result only)

FIGURE 23: Scale of maximum Frangi score along ground truth

that particular reported width.

Fig. 24 is a recontextualization of Fig. 23, where the total number of pixels (rather than a ratio) is considered for the same sample. We see that many more vascular pixels are identified as having width 7, and that in general we cannot provide a completely faithful mapping of reported vascular width to scale of largest response, but the trend of larger widths having larger scale responses and small widths having smaller scale responses remains true.

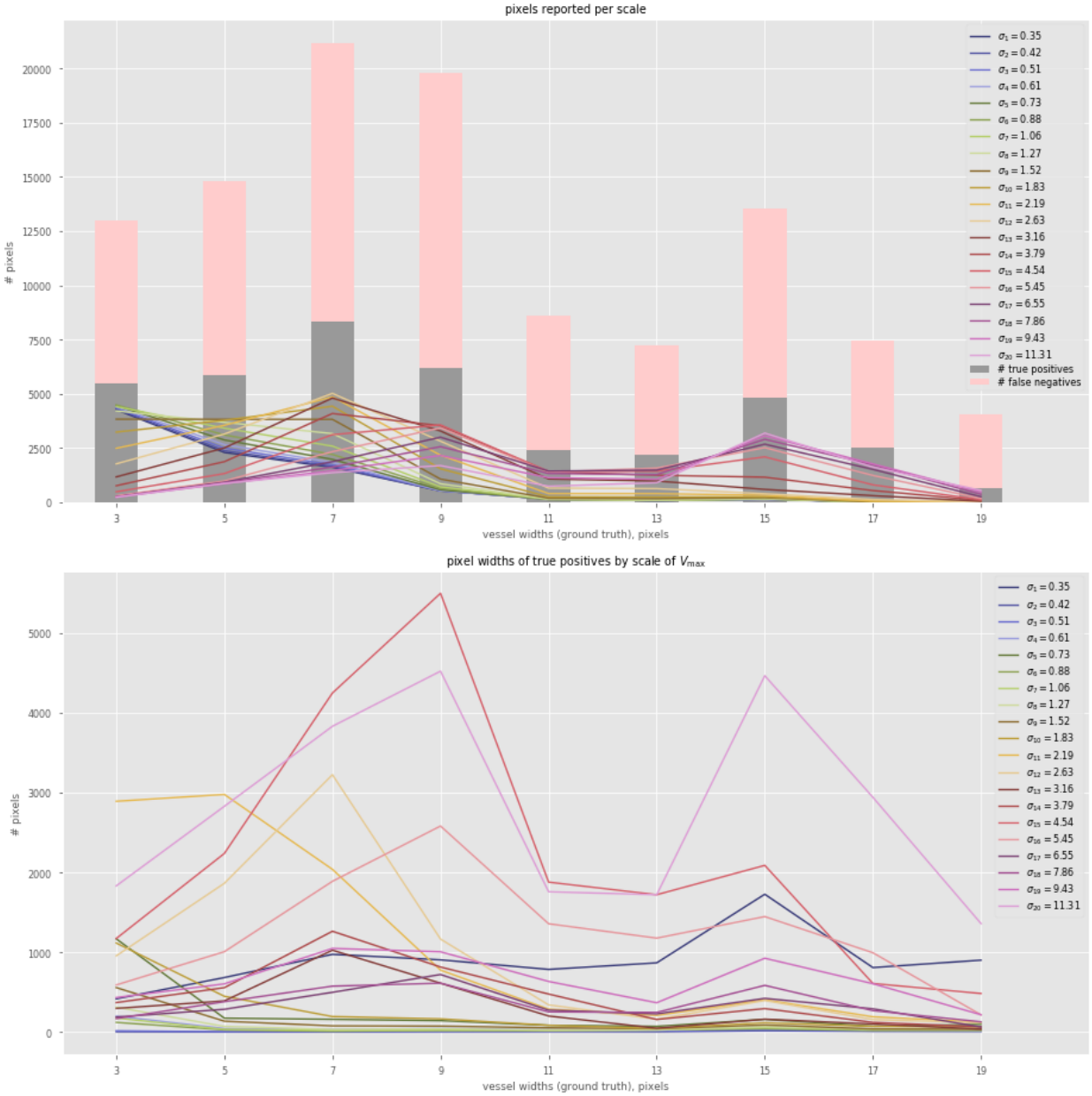


FIGURE 24: Pixel Width of Ground Truth vs. Scale Length for True Positives

CHAPTER 9

SEGMENTATION

We discuss how we use the Frangi as a prefilter and discuss several different segmentation strategies based upon our Frangi-filtered result. We will compare these methods to an unrelated segmentation strategy, the ISODATA threshold. First, we define some standard quantitative measures of success for segmentation methods.

9.1. Binary classifications, scoring methods, and the confusion matrix

We wish to come up wish a means of gauging the success of an arbitrary segmentation and will adopt a binary classification model to do so. We end up with a boolean matrix the size of the image, and compare it to the ground truth, the trace, and then compare them to get the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). We demonstrate these four labels visually with a confusion matrix, as in Fig. 25.

Although there are many measures to gauge the success of binary classification, we will focus on two that we find particularly illustrative. The first is precision, given by

$$\text{precision} = \frac{TP}{TP+FP} \tag{9.1}$$

and the second is the Matthews Correlation Constant (MCC), given by

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{9.2}$$

where precision is a ratio between 0% and 100% and the MCC is a measure between -1 and 1. Precision (or positive predictive power) is defined as the ratio between how many pixels were labeled correctly (true positives) and all pixels labeled positive (either correctly or

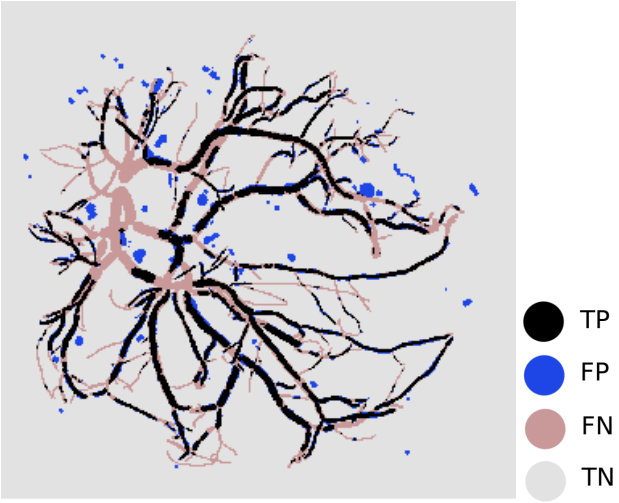


FIGURE 25: Sample confusion matrix

incorrectly). This is a useful score for us–if we are using Frangi as a prefiltering for a more robust technique, then we would rather provide an incomplete starting point, rather than an inaccurate one. Precision therefore does an okay job of representing that scenario: we are not penalized for what we do not label as true, as long as our reports of true are correct.

Of course, we cannot rely on precision as our sole quantitative factor alone–we could simply return everything negative and receive a perfect score of 100% precision. Therefore we will gauge that measure with that of the MCC [29]. The main advantage of the MCC is that it is well balanced no matter what the size of the two classes are, and will only be high if the approximation scores well against both labels. A score of 1.0 means the approximation is 100% correct, a score of -1.0 means that everything is completely incorrect, and a score of 0 means that the test performs only as well as random guessing. In our analysis, we will consider both the MCC and precision of a particular segmentation simultaneously. We would like an MCC score as high as possible, but will contextually settle for a lower score as long as the approximation is sufficiently precise.

One final point about these measures is that we have decided to report their scores only within the placental plate, rather than the entire rectangular image. Since the area outside of plate

is masked from consideration, those points will be true negatives no matter what, and we don't want these points to artificially inflate our score. That being said, we do currently concede one part right now: we will also mask an area around the umbilical cord insertion point, as the large amount of noise here will mean that our scores are artificially low. We would like to remove these areas, but for now we will simply not score them.

9.2. Postprocessing Techniques

Here we develop four relatively straightforward methods of postprocessing the multiscale Frangi output to obtain an actual PCSVN extraction. Again, we stress that the Frangi filter itself does not produce a segmentation itself. In fact, Frangi in his original paper [12] refrained from any explicit analysis of the Frangi score apart from taking the maximum across all scales. Still, we wish to demonstrate some immediate methods of postprocessing these samples in order to illustrate the usefulness of this optimized Frangi filter.

9.2.1. Method A: Fixed Threshold

Like [2], we consider a thresholded $\mathcal{V}_{\max}$ as a crude segmentation. In the fixed threshold method, we choose some threshold α and we say that a pixel (x, y) of the image corresponds to a vessel if $\mathcal{V}_{\max} > \alpha$. This α, as noted above, is unfortunately highly dependent on the image domain, and this merging method will tend to happily allow noise generated from scales that are too large or too small if Σ is not chosen correctly. We would hope that some normalization of our data set would permit a single choice of α across all samples, but unfortunately we cannot seem to guarantee this. Without prior knowledge of an appropriate choice of α, we may have to simply select α by trial and error.

Another issue with this is the individual scales of the Frangi filter in the extreme cases are not known to scale–although Lindeberg introduced a normalization factor based on the scale to apply to the derivatives, we do not know of an optimal factor to use.

Unfortunately, even with our "rescaled" Frangi filter, this α cannot be picked without regard for the particular image domain. Equally problematic, we cannot guarantee that the Frangi

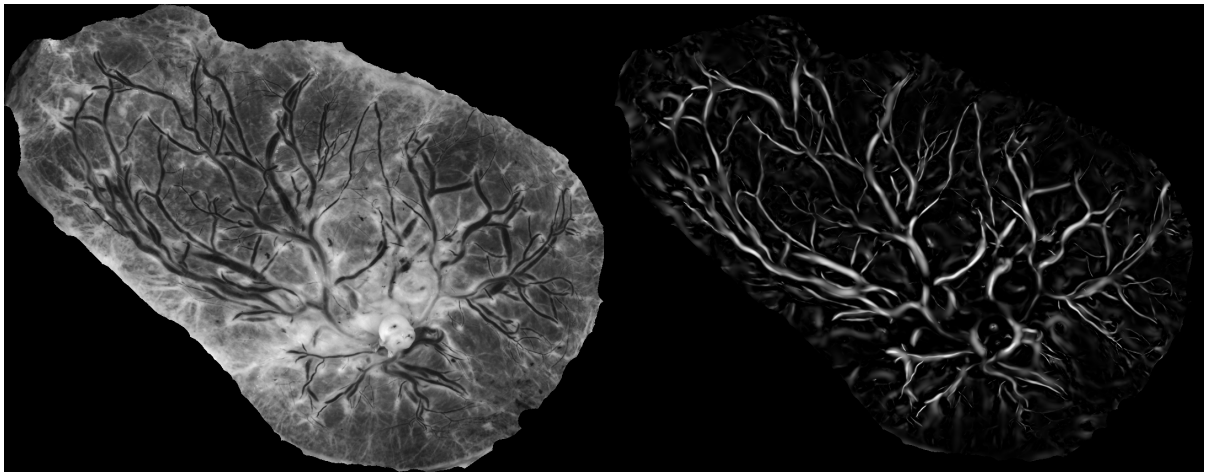
filter will decay as our scale exceeds the the bounds where structure is expected to be found.

If we insist on such a performing such a thresholding, the "correct" choice of α unfortunately seems to depend on the image domain, so user intervention when dealing with the problem domain seems to be the best strategy. A more sincere use of thresholding might be to choose a relatively high threshold, and then use the result for a further technique. We will discuss alternatives methods of aggregating results from our multiscale method, as well as optimal values for parameters and scales. As a final note, we admit that any future extensions of this work should not hold too much stock in this thresholded result. Analyzing the raw vesselness score Eq. (5.1), or even the un-merged scale-wise scores Eq. (5.3), would be far more rewarding.

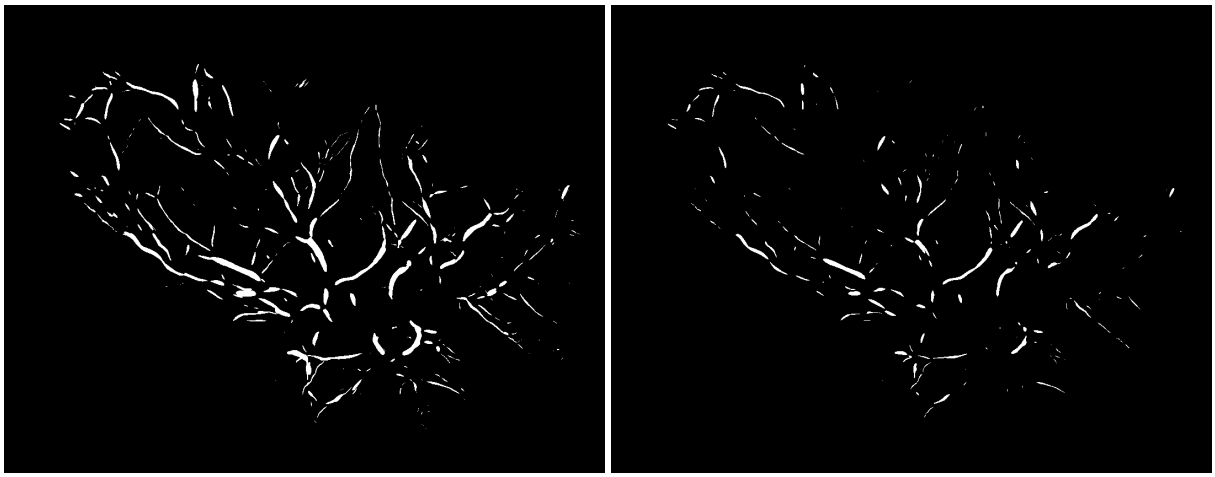
9.2.2. Method B: Scalewise Percentile Based Merging

There is an alternative method of thresholding which avoids the domain-dependent selection of a fixed threshold α altogether. Instead, we could simply select the highest scores from each responses: we calculate a high percentile score and threshold at that value. Due to the large number of zeros outputted by the filter, we opt instead to take the pth percentile of only nonzero values of each $\mathcal{V}_\sigma$. We briefly demonstrate this in Fig. 26 on a particularly well-behaved sample. The idea behind percentile-based merging is beneficial for multiscale methods. At each scale, we would like to assume that there is *some* curvilinear content that could be identified. With that in mind, we could simply accept from each scales scores in a very high percentile. We chose for our demonstration a fairly large percentile, 95, and furthermore bolster this by requiring that any selected pixels be in the 95th percentile of nonzero and unmasked pixels–otherwise the average is artificially low due to the large background and pixels with zero Frangi score. The use of percentiles removes dependence of picking a particular threshold on the problem, while allowing the most prominent features to emerge at each scale

Of course, the downside of this method is that we do not know in general the size of the network, and we may miss entire branches of the vascular network if there is a large amount of curvilinear content elsewhere in the image that is simply less prominent than some other features



(a) grayscale image (b) $\mathcal{V}_{\max}$



(c) scalewise percentile filtering ($p = 95$) (d) scalewise percentile filtering ($p = 98$)

FIGURE 26: Nonzero-percentile thresholding of $\mathcal{V}_{\max}$ (95th and 98th percentile)

.

which are prominent at the same scale. An additional caveat of this method is that, since all scales are given equal importance, this method could introduce a large amount of noise if the range of scales probled Σ, in particular the extremes $\sigma_{\min}$ and $\sigma_{\max}$ are not chosen carefully. That being said, the avantage of this method is that smaller scale features are more reliably extracted, whereas the slightly weaker Frangi signal at smaller scales causes smaller scale phenomena to less likely show up in a fixed threshold extraction.

9.2.3. Method C: Scale-Based Random Walker

We observed that areas where Frangi scores are zero in well-behaved samples seem to neatly outline prominent vascular features. Following this idea, we employed a random walker segmentation [30] (which is implemented by `scikit-image`). Random walk segmentation comes about by solving a diffusion problem over a discrete array (in this case, the Frangi vesselness score itself) given starting markers. At each scale, we positively labeled pixels whose Frangi score was very high ($\mathcal{V}_\sigma(x_0, y_0) > .4$), and negatively labeled pixels whose score was 0 (i.e. where the leading principal eigenvalue was positive). The result of this technique is demonstrated in Fig. 27 and the result (along with the original sample for comparison) is shown in Fig. 28. In Fig. 27, the first column is the Frangi vesselness score at that scale, where black is a score of 0, to emphasize the difference between a score of zero and even a very small positive score, which appear in blue. The middle score are markers passed to the random walker–blue are seeds labelled with a "1" (where the Frangi vesselness score is 0), green is labeled "2" (where $\mathcal{V}_{\max} > .4$), and purple represents unknowns that will either assigned either label. In the last column, the result of the random walker is given–areas that have been added to the label "2" are shown in yellow. Although the result of random walker segmentation is technically a binary matrix, we still show the original seeds of label 2 in green for easier comparison. Similarly, the purple in the right column has actually been labeled "1" for non-vascular, but is left in its original color to emphasize what was assigned background. In Fig. 28 we show the original image and the result of merging all positively marked pixels at each scale. Black means the pixel was unmatched, while increasing

colors of blue (larger scales) to white (smaller scales) indicate the smallest scale from which a pixel was marked after the random walker technique. Though we shall set up the multiscale method slightly differently in Chapter 8, we used a Frangi anisotropy coefficient of $\beta = 0.35$, and 12 scales logarithmically spaced from $\sigma_1 = 2^{-1.5}$ to $\sigma_{12} = 2^{3.5}$ to generate these figures. There is a parameter for the random walker algorithm (unfortunately also called β) which serves as a diffusion penalization coefficient (larger values making diffusion over the image less likely). We used `scikit-image`'s default value of 130.

9.2.4. Method D: Trough Filling

As we discussed in Section 5.2, we can simultaneously calculate the Frangi filter for light and dark curvilinear structures without any added computation time. We shown the signed result of the Frangi filter at different scales two examples in Fig. 29 and Fig. 30, we can simultaneously calculate calculate the Frangi filter response for bright curvilinear features (dark background) and dark curvilinear features (dark background). These two figures are the analogues to Fig. 6 and Fig. 7. Since the Frangi filter normally throws away any response where $\lambda_2 < 0$ (if dark curvilinear features are targeted) or $\lambda_2 > 0$ (if light curvilinear features are targeted), we lose no computation time at all by simply keeping both, although we must store more results. After computing the multiscale result, we can easily separate these into a positive and a negative strain, which we will denote $\mathcal{V}_\Sigma^{(+)}$ and $\mathcal{V}_\Sigma^{(-)}$, and then merge each as we would with a traditional multiscale Frangi filter, giving us $\mathcal{V}_{\max}^{(+)}$ and $\mathcal{V}_{\max}^{(-)}$. That is, our $\mathcal{V}_{\max}^{(+)}$ is the same as our $\mathcal{V}_{\max}$ in a conventional Frangi filter, and $\mathcal{V}_{\max}^{(-)}$ is the same result as if we had taken the Frangi filter while only looking for the opposite type (light/dark) curvilinear feature. Plotting $\mathcal{V}_{\max}$ over a scale of $[-1, 1]$ demonstrates an interesting effect. Whereas the Frangi filter generally is not reliable in terms of accurately predicting widths, we *can* get a sense of the width by looking at where there is a relatively strong response of opposite sign.

That is, at the "foot" of every trough on either side, we can see a bordering curvilinear structure of opposite sign. We perform strict Frangi filtering and separate the positive and

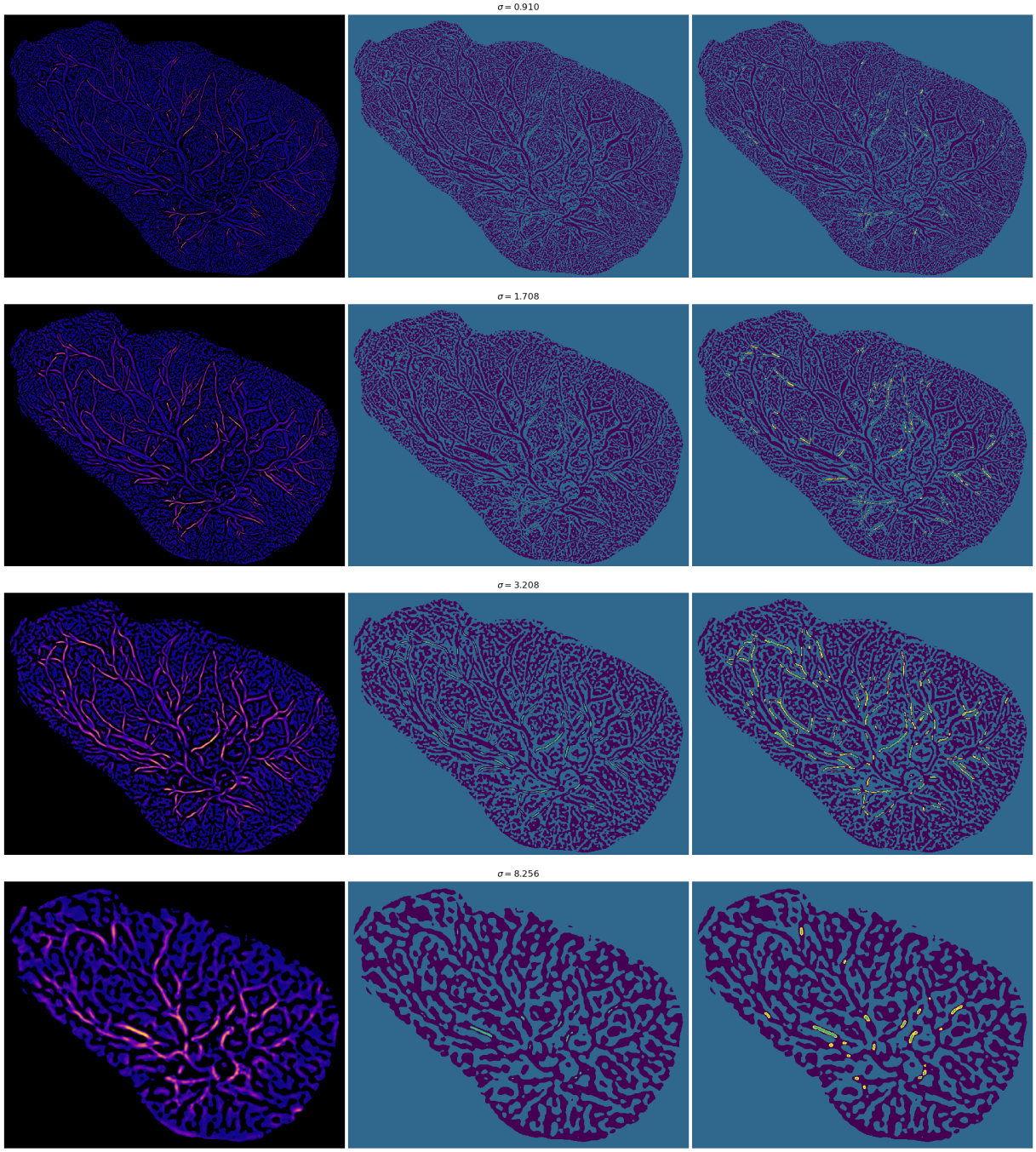


FIGURE 27: Scale-wise random walker segmentation (select scales)

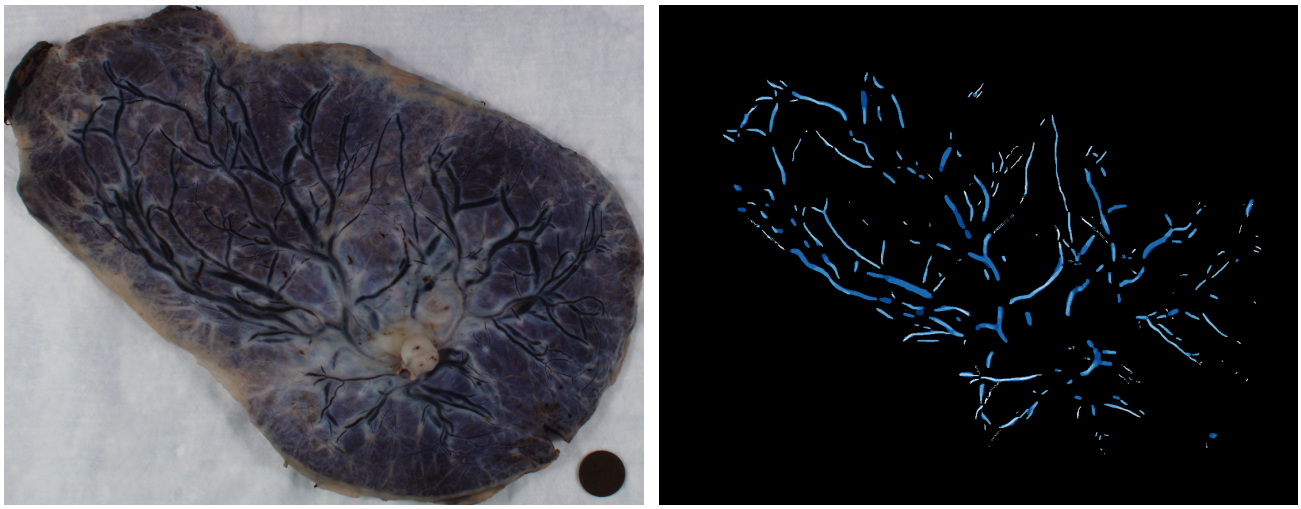


FIGURE 28: Random walker segmentation (sample and merged result)

negative components. We then perform a different threshold for each signed portion of the Frangi response–a strict one (say $\alpha^{(+)} > .3$) for the conventional $\mathcal{V}_{\max}$, and a much looser one for our opposite signed $\mathcal{V}^{(-)}_{\max} > \alpha^{(-)} = .01$. We also truncate the scales we consider for calculating $\mathcal{V}^{(-)}_{\max}$, considering only the 6 smallest scales of 20, since we empirically notice that the bordering curvilinear features are consistently narrower than the vessels themselves.

We will refer to any pixel where $\mathcal{V}^{(+)}_{\max} > \alpha^{(+)}$ as "in the trough" (assuming dark curvilinear features), and any pixel where $\mathcal{V}^{(-)}_{\max} > \alpha^{(-)}$ as potentially "on the lip" of the trough. The process of "trough-filling" then proceeds as follows. For each pixel within the trough, we iterate over disks of integer radius and dilate the pixel by a disk of that radius if that disk includes a point on the lip (where $\mathcal{V}^{(-)}_{\max} > \alpha^{(-)}$. This allows us to extend the Frangi output from the point where it's strongest all the way to the base of the curvilinear structure, providing a much better indication of the true width of the curvilinear structure.

In Fig. 31 we demonstrate the process of creating a trough dilation segments. The top left is the original sample. This was a $N = 20$ Frangi filter with log range from -1.5 to 3.2 and Frangi parameters $\beta = 0.15$ and $\gamma = 1.0$ The middle top and top right shows $\mathcal{V}^{(+)}_{\max}$ and $\mathcal{V}^{(-)}_{\max}$ respectively. The bottom then shows the two markers $\mathcal{V}^{(-)}_{\max} > \alpha^{(-)} = 0.01$ (in red) and $\mathcal{V}^{(+)}_{\max} > \alpha^{(-)} = .3$ (in

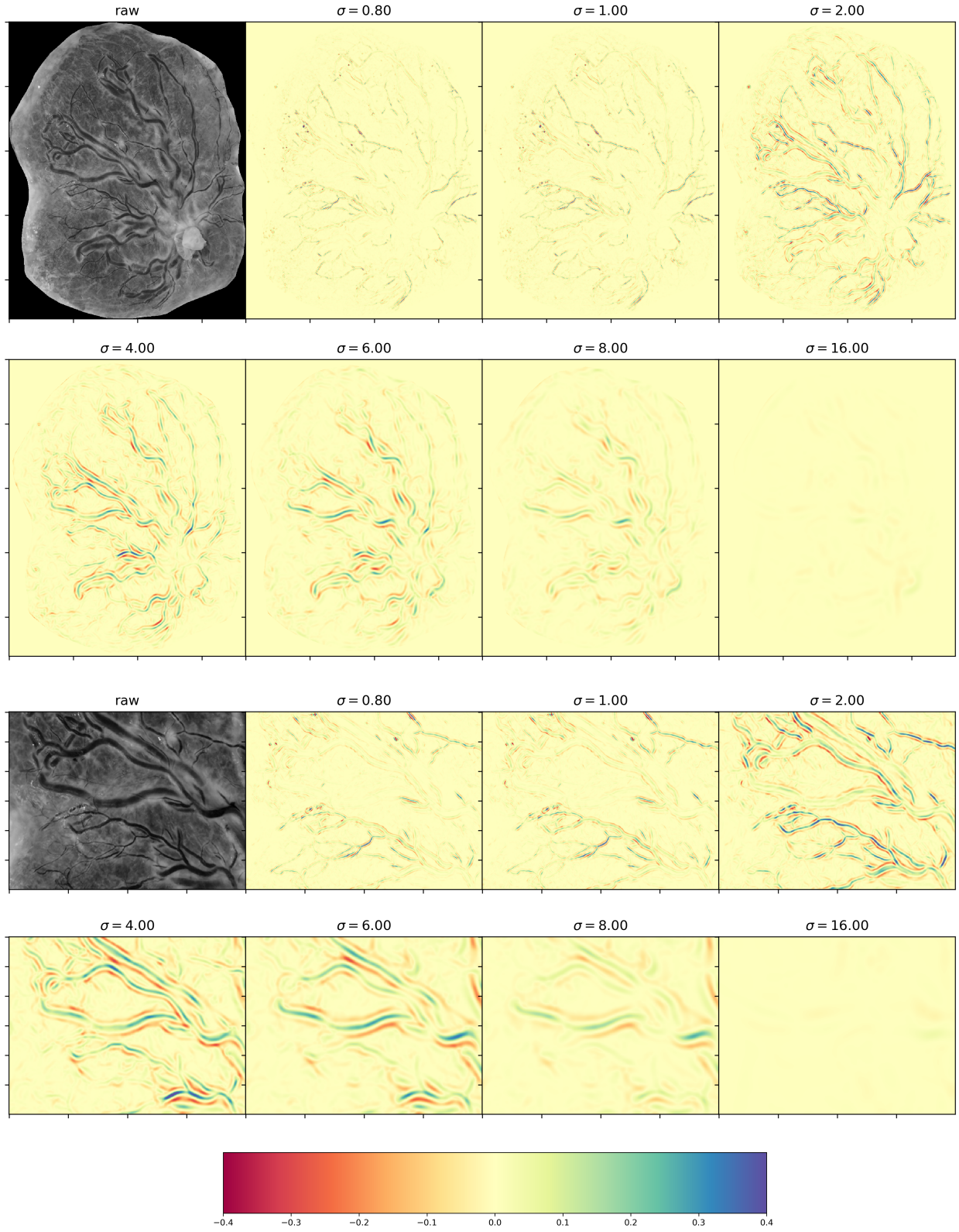


FIGURE 29: Signed Frangi output (plate and inset) (Example 1)

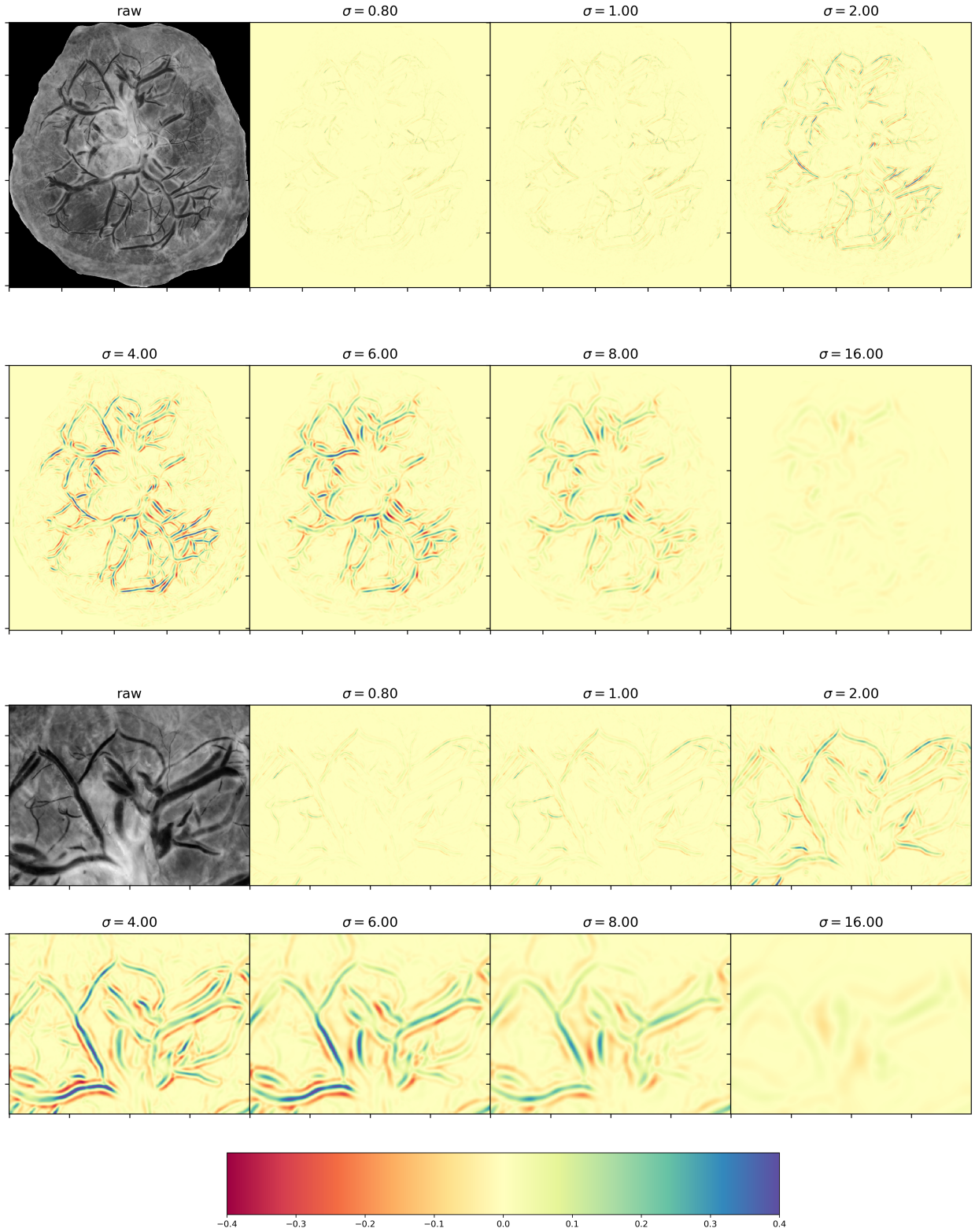


FIGURE 30: Signed Frangi output (plate and inset) (Example 1)

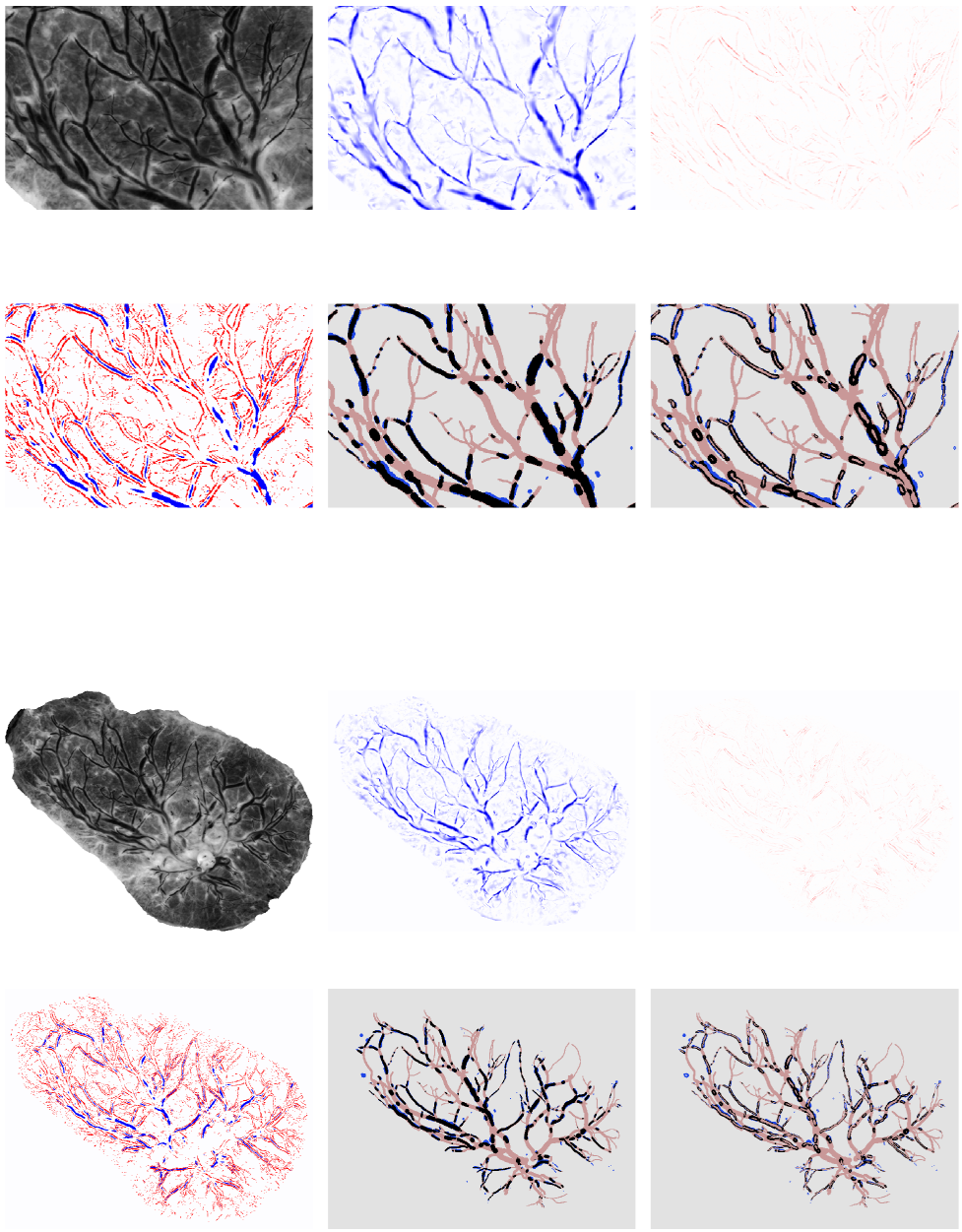


FIGURE 31: Trough dilation process (plate and inset)

blue). We build $\mathcal{V}_{\max}^{(-)}$ only using the 6 smallest scales, since the lighter curvilinear 'lip' of the trough is of smaller radius than the radius of the trough itself. Probing at too large scales could perhaps introduce noise. We will call this subset $\Sigma^{(-)} \subset \Sigma$ and choose $\Sigma^{(-)} = \{\sigma_1, ..., \sigma_6\}$.

9.3. Nonfrangi Segmentation Methods

Of course, there are many other viable methods of segmentation. We refer to [31] for several high-performing (albeit considerably resource intensive) segmentation methods on this dataset, namely involving shearlets, Laplacian eigenmaps, and a conditional generative adversarial network.

We will be content in the present paper to simply compare our results to a relatively fast technique that is not based on differential geometry or morphology, but instead that of a global threshold on the grayscale image. Although many viable thresholding algorithms exist, we opt for an intermeans threshold, or the Ridler-Calvert or ISODATA method, as described in [32], which is implemented in `scikit-image` as `filters.threshold_isodata`. This functions uses an iterative process to find the smallest threshold which is midway between the mean intensities of the high pixels and the low pixels. In other words, the optimal α_{ISO} satisfies

$$\alpha_{\text{ISO}} = \arg\min_{\alpha} \left(\frac{1}{2} \left[\text{mean} \{ \mathtt{I}(x,y) \mid \mathtt{I}(x,y) \leq \alpha \} + \text{mean} \{ \mathtt{I}(x,y) \mid \mathtt{I}(x,y) > \alpha \} \right] \right) \tag{9.3}$$

Since the vascular structure in our image domain is darker than the background, we select pixels where $\mathtt{I}(x,y) < \alpha_{\text{ISO}}$.

9.4. Comparison of Segmentation Methods

In our demonstration of segmentation methods across all 201 samples in our image domain, we use the preprocessing procedure described in Chapter 7.

Our multiscale Frangi filter setup involves 20 scales spaced logarithmically from $\sigma_1 = 2^{-1.5} \approx .35$ to $\sigma_{20} = 2^{3.2} \approx 9.20$. We will compare the results of 6 different segmentation

methods using three different parametrizations of the Frangi filter. These parametrizations and segmentation methods with specific parameter choices are summarized in Tables 8 and 9.

Label	Description	Parameter(s)
thresh-high	Fixed threshold of Frangi filter $\mathcal{V}_{\max} > \alpha$	$\alpha = 0.3$
thresh-low	Fixed threshold of Frangi filter $\mathcal{V}_{\max} > \alpha$	$\alpha = 0.2$
snz-p-high	scalewise nonzero percentile filtering of $\mathcal{V}_\Sigma$	$q = 95$
snz-p-low	scalewise nonzero percentile filtering of $\mathcal{V}_\Sigma$	$q = 98$
TF	trough-filling method	$\alpha^{(+)} = 0.3$, $\alpha^{(-)} = .01$, $\Sigma^{(-)} = \{\sigma_1, \cdots, \sigma_6\}$
ISODATA	Non-Frangi global threshold	see Eq. (9.3)

TABLE 8: Summary of segmentation methods

Label	β	γ
standard	0.5	0.5
semistrict	0.15	0.5
strict	0.15	1.0

TABLE 9: Summary of Frangi parametrizations for segmentation demo

In Fig. 32, we look at the results of our various segmentation procedures for a typical well behaved sample. We can see that there are fewer false negatives for strict parametrization than with a standard parametrization. Even though the best MCC score across the 10 demonstrated in Fig. 32 is achieved by trough filling under standard parametrization, we note that trough filling in the strict parametrization has comparatively fewer false negatives and much higher precision. The highest precision is achieved by our higher fixed threshold. Thus, we should see the high MCC score achieved by the trough filling method as a testament to the usability of a simpler threshold (such as FT-high) as a precursor to more complete segmentation methods.

In Fig. 34 we graph the MCC and precision scores across our entire set of 201 images. In each boxplot, the median is labeled, with first and third quartiles making up the edges of the box. The ends of the whiskers represent $median \pm 1.5 * (Q3 - Q1)$. The quantity $(Q3 - Q1)$, the difference of the third quartile (where $p = 75$) and first quartile (where $p = 25$) is the so-called

interquartile range [26]. Outliers outside of those whiskers, from samples such as those in Fig. 19, are plotted as individual data points.

Across all samples, maximum precision was achieved by the higher fixed threshold of $\alpha = 0.3$ in all but 22 samples, and the trough filling method offered the best MCC in about three-fourths of the samples (51 of 201). Where the trough filling method was not most accurate, we noticed that the $q = 95$ nz-percentile method had a slightly higher MCC score. From viewing the samples, we see that noise from the umbillical cord insertion point still hinders a number of samples, causing noise within the Frangi filter.

From viewing Fig. 34 we can see that, for any of the three parametrizations, we achieve a higher precision and lower MCC from increasing the fixed threshold. The same trend occurs between scalewise nz-p segmentations when we increase the quartile (from q=95 to q=98). The median MCC of the ISODATA method was lower than all Frangi-based segmentations across all with the exception of the highest fixed threshold under strict parametrization–although the latter method has the highest median precision score by far. ISODATA has a much lower precision than any of the other methods present here.

We notice a larger number of MCC outliers for scalewise methods than fixed threshold based methods. We theorize that this is due to the difficulty in knowing *a apriori* an appropriate range of scales to probe. Future work therefore could immediately improve upon the results of this chapter by finding an appropriate range or, more likely, a normalization factor for scales that makes the success of segmenation less dependent on σ_{min} and σ_{max}.

From Fig. 34 we also notice that the MCC and precision for each scalewise method is not as affected by changes in parameterization as the fixed methods. We suggest that is because stricter parameters have the effect of rescaling to some degree within each step of the multiscale method, so that the same pixels at each scale ultimately occur at each scale's highest percentiles.

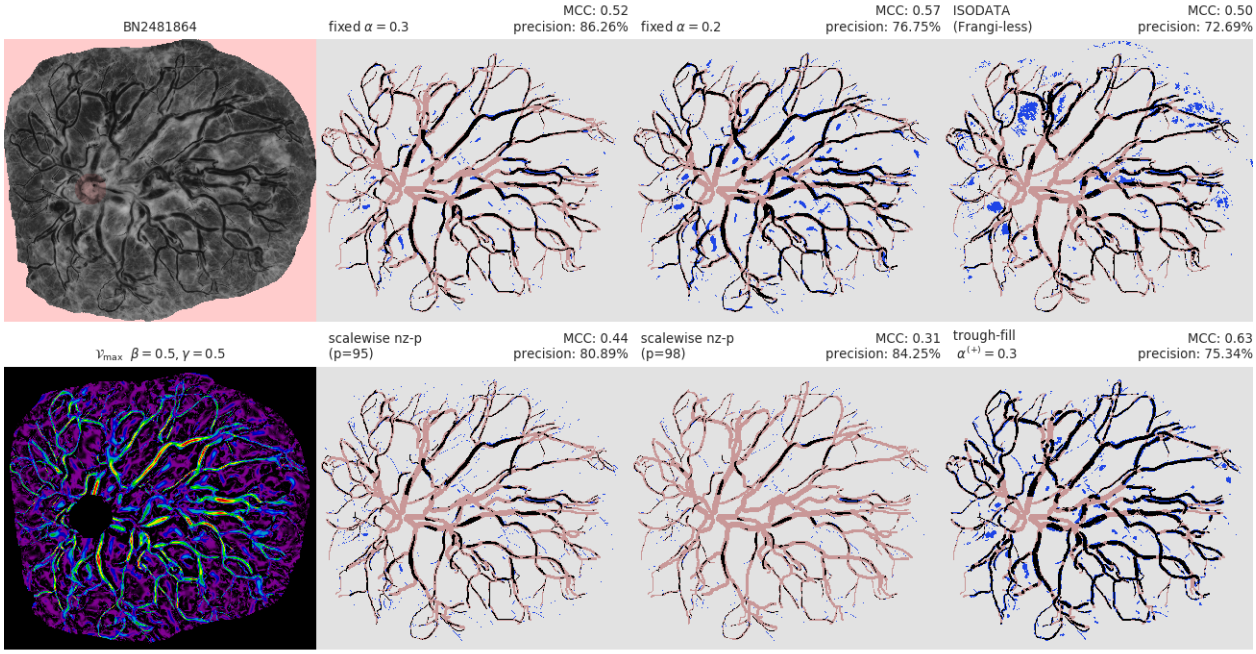
Looking at Fig. 33, we see a situation in which scalewise methods do poorly. The $p = 95$ scalewise method under either parametrization, as well as the lower fixed threshold under standard

parametrization show some curvilinear noise between vessels. These come from larger scales, so there is apparently no signficant vasculature being picked up at the largest scales, causing noise to appear. This is less pronounced for fixed thresholding using stricter parameters, although some of this noise can still be seen in the strict Frangi's $\mathcal{V}_{\max}$ plot.

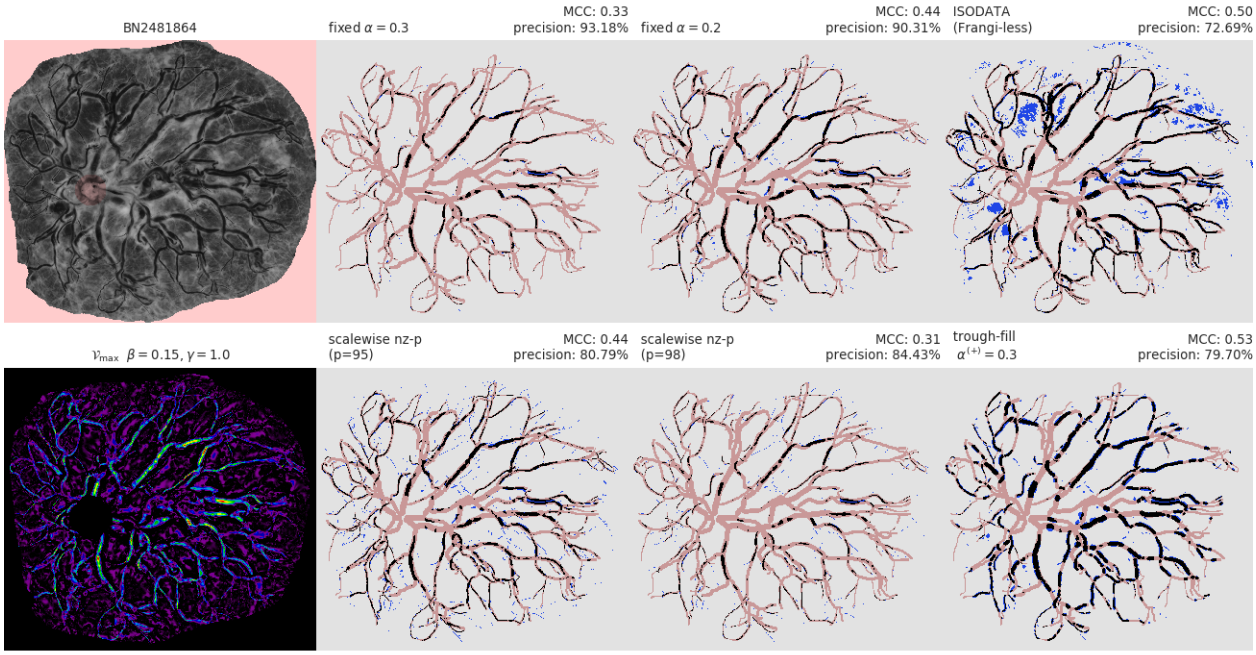
9.5. Network Completion

We can see from looking at the $\mathcal{V}_{\max}$ outputs of our samples that even where gaps exist in the segmentation output, there is still *some* Frangi output where the gap exists. For various reasons, the output might not be as strong at that particular region, but we can use that fact to bridge some of these gaps. To do so would be to achieve "simple gap completion"–that is, connecting the vascular network to compensate for the shortcomings of a particular segmentation method. We differentiate this from "predictive gap completion," which we will use to signify the problem of connecting the vascular network where the tracer themselves had to guess what was happening with the network. For example, the entire area surrounding the umbilical cord insertion point in Fig. 33 is difficult to make out. The tracer ultimately used some knowledge of vascular network growth to predict what was happening in this region to produce the ground truth. Predictive gap completion would also need to be used to solve crossings of veins and arteries.

First, once we've produced a segmentation, we use morphological thinning to reduce it to one pixel width with [33] and look for endpoints of an otherwise connected point. Using a 3×3 structuring element, we iterate over each pixel and identify how many local neighbors it has. If a pixel has zero or one local neigbors, we identify it as an endpoint of the partial network. After identifying these endpoints, we assign each a label (i, j) depending on where the neighboring pixel is located, as according to Fig. 35. We deem two endpoints potentially connectible only if they're not connected on the same side. That is, their labels (i, j) and (i', j') must have $i \neq i'$ or $j \neq j'$ (unless i or j is 1). For example, if an endpoint is connected to the partial network on its top side, any endpoint that connects to it cannot also be connected to a network on its top. If a pixel has no connections at all, with label $(1, 1)$, we do not restrict its connetions at all. To save time (though

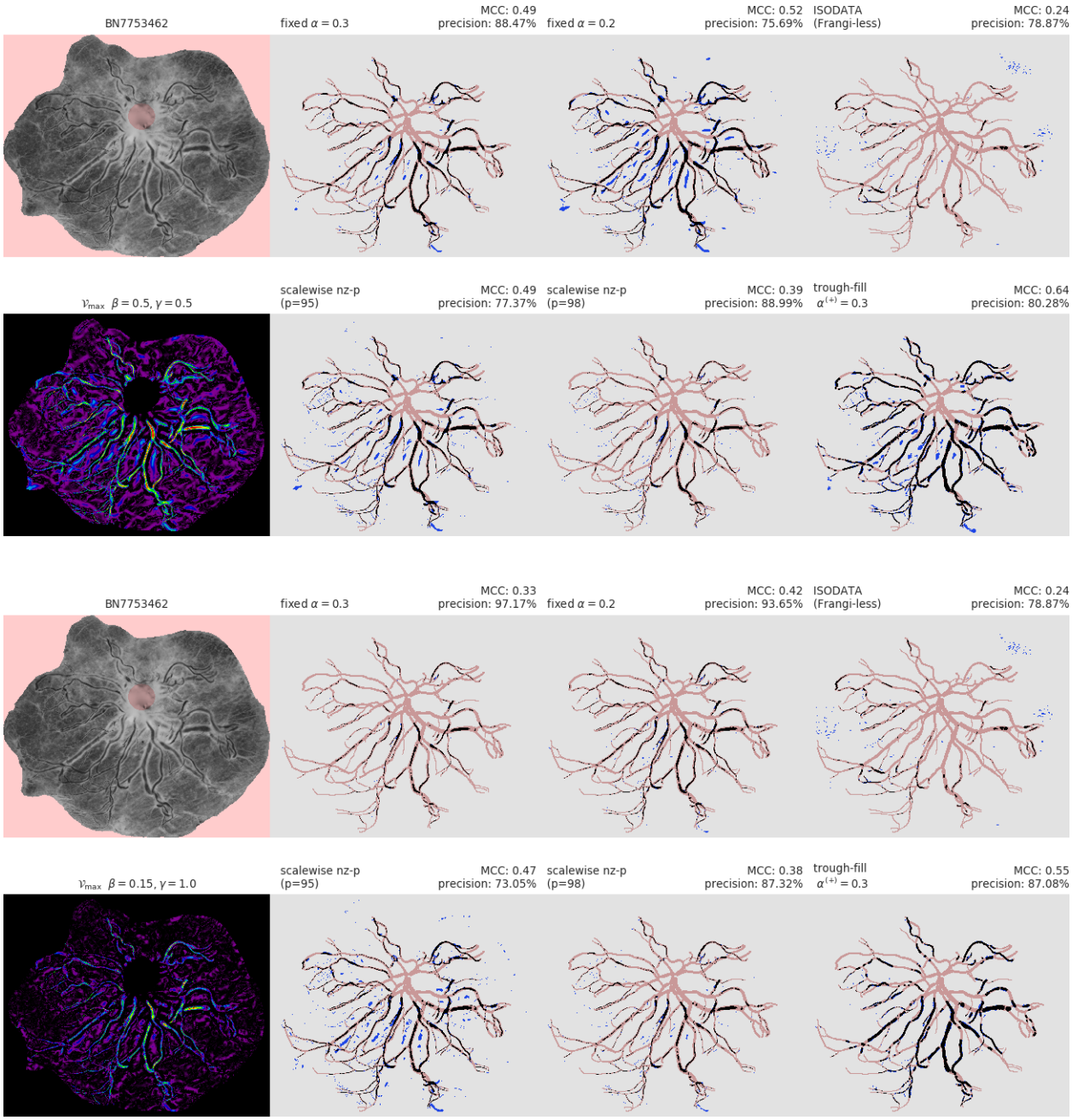


(a) standard parametrization



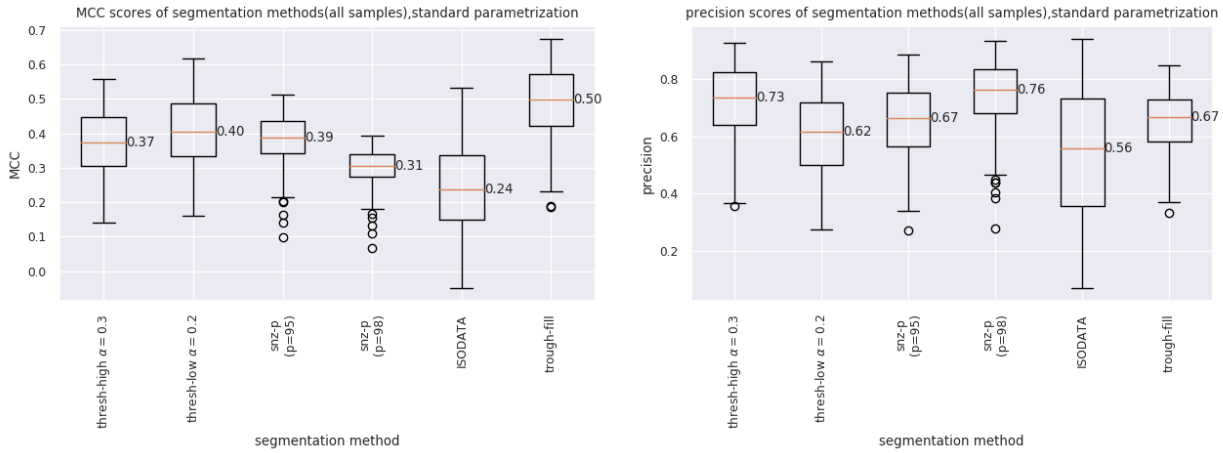
(b) strict parametrization

FIGURE 32: Segmentation results, example 1 (standard and strict parametrization)

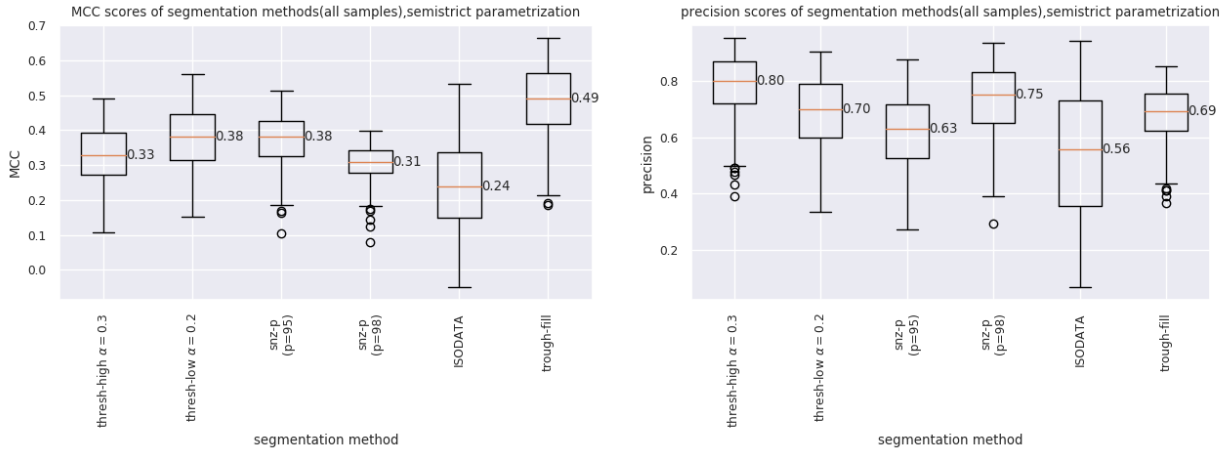


(b) strict parametrization

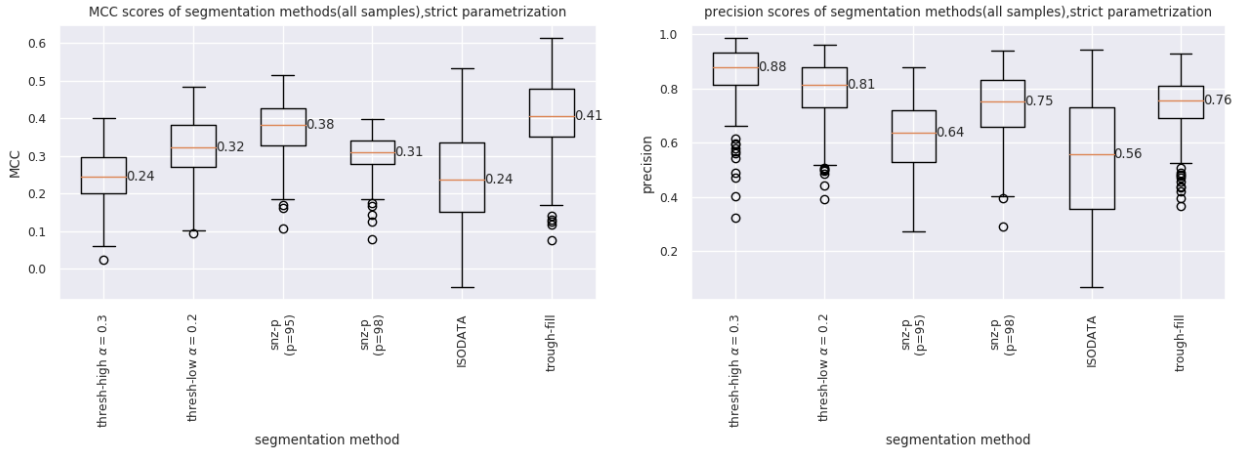
FIGURE 33: Segmentation results, example 2 (standard and strict parametrization)



(a) standard Frangi parametrization ($\beta = 0.5, \gamma = 0.5$)



(b) semistrict Frangi parametrization ($\beta = 0.15, \gamma = 0.5$)



(c) strict Frangi parametrization ($\beta = 0.15, \gamma = 1.0$)

FIGURE 34: MCC and precision of segmentation methods (201 samples)

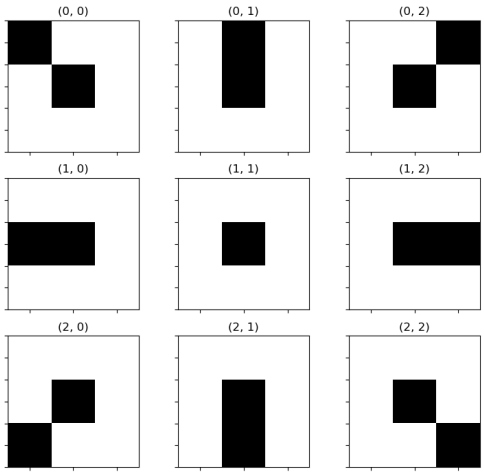


FIGURE 35: Endpoints labels based on adjacent neighbor location

we don't anticipate it will affect the result much), we also restrict pais from being more than a set distance away (in this case, 100 pixels in Euclidean length).

After we limit the connections, we consider each pair of endpoints and draw a straight line segment between the two. If that passes through a point where $\mathcal{V}_{\max}$ is 0, we disallow that pair as well. We also disallow any line which crosses any part of the network which is known to exist.

Finally, from the list of all remaining pairs of endpoints, we simply select the path along which the maximum mean value of $\mathcal{V}_{\max}$ is achieved. Fig. 36 shows non-violating paths between end-point pairs. The coloration shows the average value of $\mathcal{V}_{\max}$ along any straight path between compatible endpoints for which there does not occur any pixel with zero $\mathcal{V}_{\max}$. From these we can choose the path with largest average $\mathcal{V}_{\max}$. This partially completed network is shown in Fig. 37(in yellow) overlaid on $\mathcal{V}_{\max}$. We show this result as an indication of what can be done with the simple generation of $\mathcal{V}_{\max}$, although we simply demonstrate it here rather than including it in our main segmentation results. This is because, for its complexity, we hope we can instead solve both simple and predictive gap completion using a single method.

Finally, one final comment we can make. For the scale the maximum Frangi output occurs,

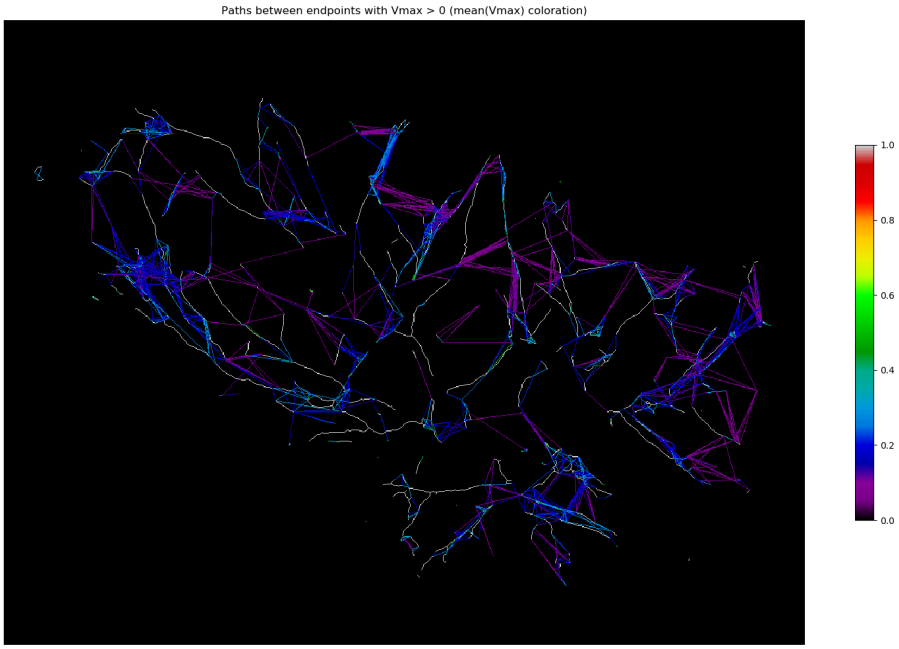


FIGURE 36: All lines between endpoints with nonzero $\mathcal{V}_{\max}$

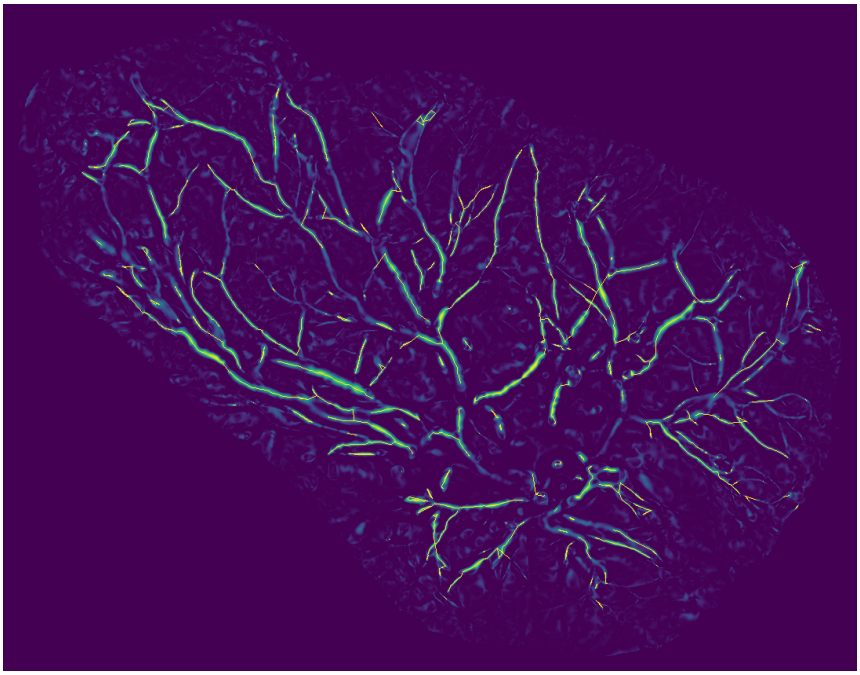


FIGURE 37: Partially completed network

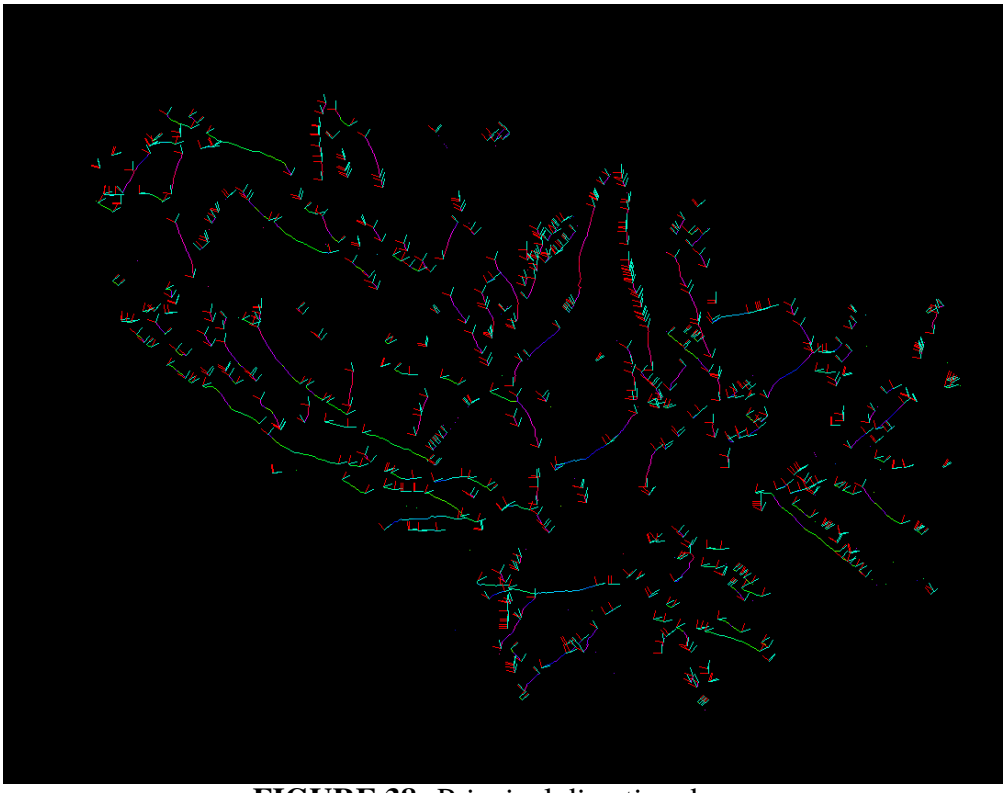


FIGURE 38: Principal direction demo

at, we can look at the Hessian matrix itself to glean additional information. Recalling our discussion from Chapter 2, we can instead of calculating the leading eigenvalue, we can calculate the leading *eigenvector*. Just as the leading eigenvalue of the Hessian is a good approximation of the principal curvature at that point, the leading eigenvector is a good approximation of the principal direction. In Fig. 38 we show the vectors of the leading (and trailing) eigenvectors of each endpoint of the thinned, approximated network. We note that these could additionally be used towards the goal of network completion, but again we hesitate and instead look for a better overarching method that would address the gap problem.

9.6. Further Directions

Future research will first aim toward automating more of the preprocessing, specifically toward identifying the perimeter, umbilical cord insertion point, and any cuts without relying on a manual trace. As mentioned in Chapter 7, a more careful preparation of samples (i.e. as the picture is taken) would alleviate some of the difficulty of image registration. We also should improve our glare reduction algorithm, as it currently relies on an arbitrary threshold. We also need to remove the umbilical cord stump, as a large amount of noise around that point is still causing a large amount of noise in many samples. As far as our multiscale method goes, we would like to develop a notion of automatic scale selection, that would help us better normalize smaller scales, as well as allow us to find a specific largest scale (as some vessels in specific samples were only easily identified at very large scales, whereas these scales would introduce only noise in most other samples).

CHAPTER 10

CONCLUSION

We justified the use of differential geometry in 2D discrete image processing to effectively re-establish the usefulness of the Frangi filter for identifying curvilinear content in digital images. We vastly improved upon the implementation of the Frangi filter and provided several modifications of the Frangi filter–namely to normalize the filter's output each scale to allow us to choose a structureness parameter more freely. Our improved implementation allowed us to calculate our result faster, incorporating many scales. We also rationalized choosing stricter parameters for our particular agenda. We used our multiscale Frangi vesselness measure to suggest several alternative approaches at merging the vesselness and compared their effectiveness as a precursor to segmentation and eventually network completion.

Any additional research on this problem that wishes to use the Frangi filter as a prefilter (e.g. the trough completion method detailed in Chapter 9 or some more involved algorithm) will likely need to solve the network completion problem, which we addressed briefly at the end of Chapter 9.

APPENDICES

APPENDIX A

CODE LISTINGS

The following python scripts and modules were developed with the following packages:

- `python 3.6`
- `numpy`, version 1.12.0
- `scipy`, version 0.19.0
- `scikit-image`, version 0.13.0
- `matplotlib`, version 2.02

Earlier versions of these packages may be compatible but are not guaranteed to be so. All scripts used in the preparation of this thesis are available at `github.com/wukm/pycake`, although the sample images are from a private database.

We present a top level script, used to generate the segmentation examples and data for Chapter 9, is presented below. Please refer to the repository above for details of any support functions used within this script.

listings/compare_segmentations.py

```
#!/usr/bin/env python3

"""
this is the main result in which we compare the segmentations across all
samples. rerun this code changing beta & gamma to get outputs for different
segmentations.

segmentation strategies used here are:

for a fixed gamma, beta, and range of sigma
1. ISODATA (Frangiless)
2. Frangi simple threshold (high thresh)
3. Frangi simple threshold (low thresh)
4. Frangi nz-percentile (high percentile)
5. Frangi nz-percentile (lower percentile)
6. trough-filling
"""

import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from placenta import (get_named_placenta, list_by_quality, cropped_args,
                      mimg_as_float, open_typefile, open_tracefile,
                      strip_ncs_name, list_placentas, measure_ncs_markings,
                      add_ucip_to_mask)
```

```
26
27 from frangi import frangi_from_image
28 from merging import nz_percentile, apply_threshold
29 from plate_morphology import dilate_boundary
30 import os.path
31 import os
32
33 # from scipy.ndimage import distance_transform_edt as edt
34 # from skimage.filters.rank import enhance_contrast_percentile as ecp
35 from scoring import confusion, mcc
36
37 from skimage.color import grey2rgb
38
39 from skimage.filters import threshold_isodata
40 from postprocessing import dilate_to_rim
41 from preprocessing import inpaint_hybrid
42 import json
43
44
45 def split_signed_frangi_stack(F, negative_range=None, positive_range=None,
46                               mask=None):
47     """Get Vmax+ and Vmax- from Frangi stack
48
49     F is the frangi stack where the first dimension is the scale space.
50     transposing it would be easier.
51
52     if negative_range is given, then only scales within that range are
53     accumulated
54
55     will return Vmax(+) and Vmin(-)
56
57     should redo this so that it returns V_{\Sigma}^{(+)} and V_{\Sigma}^{(-)}
58     """
59
60     if negative_range is None:
61         negative_range = (0, F.shape[-1])
62
63     if positive_range is None:
64         positive_range = (0, F.shape[-1])
65
66     f = F[positive_range[0]:positive_range[1]].max(axis=0)
67     nf = ((-F*(F < 0))[negative_range[0]:negative_range[1]]).max(axis=0)
68
69     if mask is not None:
70         nf[mask] = 0
71
72     return f, nf
73
74
75 #quality_name = 'good'
76 #placentas = list_by_quality(0, N=2)
77
78 quality_name = 'all'
79 placentas = list_placentas()
80
81
82 #beta, gamma, parametrization_name = 0.15, 1.0, "strict"
83 #beta, gamma, parametrization_name = 0.15, 0.5, "semistrict"
84 #beta, gamma, parametrization_name = 0.5, 1.0, 'semistrict-gamma'
85 beta, gamma, parametrization_name = 0.5, 0.5, "standard"
86
87 OUTPUT_DIR = (f'output/190304-segmentation_demo_'
88               f'{quality_name}_{parametrization_name}-ucip')
89
```

```
if not os.path.exists(OUTPUT_DIR):
    os.makedirs(OUTPUT_DIR)

N_scales = 20
THRESHOLD = .3  # \alpha^{(+)}
THRESHOLD_LOW = .2  # \alpha^{(+)}
MARGIN_THRESHOLD = 0.01  # \alpha^{(-)}
NEGATIVE_RANGE = (0, 6)
log_range = (-1, 3.2)
scales = np.logspace(*log_range, base=2, num=N_scales)

mccs = list()
precs = list()
for filename in placentas:

    # get the name of the sample (like 'BN#######')
    basename = strip_ncs_name(filename)

    print(basename, '*'*30)

    # this calls find_plate_in_raw for all files not in placenta.FAILS
    # in an attempt to improve upon nthe border
    # load sample and do pre-processing
    img = get_named_placenta(filename)
    img = inpaint_hybrid(img)
    ucip, res = measure_ncs_markings(filename=filename)
    old_mask = img.mask.copy()
    img.mask = add_ucip_to_mask(ucip, radius=50, mask=img.mask)

    # load trace
    crop = cropped_args(img)
    cimg = open_typefile(filename, 'raw')
    ctrace = open_typefile(filename, 'ctrace')
    trace = open_tracefile(filename)
    bigmask = dilate_boundary(None, mask=img.mask, radius=20)

    # rotate the image so it's hotdog not hamburger
    if img[crop].shape[0] > img[crop].shape[1]:
        # and rotating it would be fix all this automatically
        cimg = np.rot90(cimg)
        ctrace = np.rot90(ctrace)
        trace = np.rot90(trace)
        img = np.rot90(img)
        crop = cropped_args(img)
        bigmask = dilate_boundary(None, mask=img.mask, radius=20)
        old_mask = np.rot90(old_mask)

    # threshold according to ISODATA threshold for a strawman
    straw = img.filled(0) < threshold_isodata(img.filled(0))
    straw[img.mask] = False  # apply the mask

    # make a MxNxN_scales matrix of Frangi outputs (where (M,N) == img.shape)
    F = np.stack([frangi_from_image(img, sigma, beta, gamma, dark_bg=False,
                                    dilation_radius=20, rescale_frangi=True,
                                    signed_frangi=True).filled(0)
                  for sigma in scales])

    # Fneg still has negative direction but these are now separated
    Fpos = np.clip(F, 0, None)  # replace negative values with 0
    Fneg = np.clip(F, None, 0)  # replace positive values with 0

    # makes Vmaxpos and Vmaxneg (could also take max of Fpos and -Fneg above
    f, nf = split_signed_frangi_stack(F, positive_range=None,
                                      negative_range=NEGATIVE_RANGE,
```

```
                                        mask=bigmask)

    # everything in the following section needs a better name :/
    spine = dilate_boundary(f, mask=img.mask, radius=20).filled(0)
    spineseeds = (spine > THRESHOLD)
    margins = (nf > MARGIN_THRESHOLD)

    # trough-filling segmentation
    approx_tf, radii = dilate_to_rim(spineseeds > THRESHOLD, margins,
                                     return_radii=True)

    # fixed threshold
    approx_FA_high = (spine > THRESHOLD)
    approx_FA_low = (spine > THRESHOLD_LOW)

    # scalewise for 95th percentile
    ALPHAS_95 = np.array([nz_percentile(Fpos[k], 95.0)
                          for k in range(len(scales))])
    ALPHAS_98 = np.array([nz_percentile(Fpos[k], 98.0)
                          for k in range(len(scales))])

    approx_PF95 = apply_threshold(np.transpose(Fpos,(1,2,0)), ALPHAS_95,
                                  return_labels=False)
    approx_PF98 = apply_threshold(np.transpose(Fpos,(1,2,0)), ALPHAS_98,
                                  return_labels=False)

    precision_score = lambda t: int(t[0]) / int(t[0] + t[2])

    # mcc, counts, precision for ISODATA
    m_st, counts_st = mcc(straw, trace, bg_mask=img.mask, return_counts=True)
    p_st = precision_score(counts_st)

    # mcc, counts, precision for fixed threshold w/o prefilter
    m_FA_low, counts_FA_low = mcc(approx_FA_low, trace, bg_mask=img.mask,
                                  return_counts=True)
    p_FA_low = precision_score(counts_FA_low)

    m_FA_high, counts_FA_high = mcc(approx_FA_high, trace, bg_mask=img.mask,
                                    return_counts=True)
    p_FA_high = precision_score(counts_FA_high)

    # mcc, counts, precision for scalewise percent filtelr
    m_PF95, counts_PF95 = mcc(approx_PF95, trace, bg_mask=img.mask,
                              return_counts=True)
    p_PF95 = precision_score(counts_PF95)

    m_PF98, counts_PF98 = mcc(approx_PF98, trace, bg_mask=img.mask,
                              return_counts=True)
    p_PF98 = precision_score(counts_PF98)

    #mcc, counts, precision for trough_fillings w/o ECP prefilter
    m_tf, counts_tf = mcc(approx_tf, trace, bg_mask=img.mask, return_counts=True)
    p_tf = precision_score(counts_tf)




    # this is just used for visualization, not very meaningful though
    #recolor_with_negative = (f <= THRESHOLD) & (nf > MARGIN_THRESHOLD)
    #fboth = f.copy()
    #fboth[recolor_with_negative] = nf[recolor_with_negative]

    sns.set(font_scale=0.8)
    fig, ax = plt.subplots(nrows=2, ncols=4, figsize=(13,7))
```

```
218
219     I = grey2rgb(img.data)  # three channels (based on img old data)
220     I[old_mask] = (1.,1.,1.)  # make old BG white not black
221     I[img.mask] *= (1.0, 0.8, 0.8)  # overlay red on ucip mask
222     #ax[0,0].imshow(img[crop], cmap=plt.cm.gray)
223     ax[0,0].imshow(I[crop])
224     ax[0,0].set_title(basename)
225
226
227     ax[0,1].imshow(confusion(approx_FA_high, trace)[crop])
228     ax[0,1].set_title(rf'   fixed $\alpha = THRESHOLD$', loc='left')
229     ax[0,1].set_title(f'MCC: {m_FA_high:.2f}\n'
230                       f'precision: {p_FA_high:.2%}', loc='right')
231
232     ax[0,2].imshow(confusion(approx_FA_low, trace)[crop])
233     ax[0,2].set_title(rf'   fixed $\alpha = THRESHOLD_LOW$', loc='left')
234     ax[0,2].set_title(f'MCC: {m_FA_low:.2f}\n'
235                       f'precision: {p_FA_low:.2%}', loc='right')
236
237     ax[0,3].imshow(confusion(straw,trace)[crop])
238     ax[0,3].set_title(f'   ISODATA\n   (Frangi-less)', loc='left')
239     ax[0,3].set_title(f'MCC: {m_st:.2f}\n'
240                       f'precision: {p_st:.2%}', loc='right')
241
242     ax[1,0].imshow(f[crop], cmap='nipy_spectral', vmax=1.0, vmin=0.0)
243     ax[1,0].set_title(r'$\mathcal{V}_{\max}$   '
244                       fr'$\beta = beta, \gamma = gamma$')
245
246     ax[1,1].imshow(confusion(approx_PF95,trace)[crop])
247     ax[1,1].set_title(f'   scalewise nz-p\n   (p=95)', loc='left')
248     ax[1,1].set_title(f'MCC: {m_PF95:.2f}\n'
249                       f'precision: {p_PF95:.2%}', loc='right')
250
251     ax[1,2].imshow(confusion(approx_PF98,trace)[crop])
252     ax[1,2].set_title(f'   scalewise nz-p\n   (p=98)', loc='left')
253     ax[1,2].set_title(f'MCC: {m_PF98:.2f}\n'
254                       f'precision: {p_PF98:.2%}', loc='right')
255     ax[1,3].imshow(confusion(approx_tf,trace)[crop])
256     ax[1,3].set_title( '   trough-fill\n'
257                       r'    $\alpha^{(+)} =$'
258                       fr'$THRESHOLD$', loc='left')
259     ax[1,3].set_title(f'MCC: {m_tf:.2f}\n'
260                       f'precision: {p_tf:.2%}', loc='right')
261
262
263
264
265     [a.axis('off') for a in ax.ravel()]
266     fig.tight_layout()
267     fig.subplots_adjust(right=1.0, left=0, top=0.95, bottom=0.,
268                         wspace=0.0, hspace=0.05)
269     plt.savefig(os.path.join(OUTPUT_DIR, ''.join(('fig-', basename, '.png'))))
270     #plt.show()
271     plt.close()
272
273     mccs.append((filename, m_FA_high, m_FA_low, m_PF95, m_PF98, m_st, m_tf))
274     precs.append((filename, p_FA_high, p_FA_low, p_PF95, p_PF98, p_st, p_tf))
275
276 runlog = { 'mccs': mccs, 'precs': precs}
277
278 with open(os.path.join(OUTPUT_DIR,'runlog.json'), 'w') as f:
279     json.dump(runlog, f, indent=True)
280
281 # get rid of sample labels
```

```
M = np.array([m[1:] for m in mccs])
P = np.array([p[1:] for p in precs])

M_medians = np.median(M, axis=0)  # what the actual medians are (for labeling)
P_medians = np.median(P, axis=0)  # what the actual medians are (for labeling)

# segmentation strategy labels
labels = [
    rf'thresh-high $\alpha = THRESHOLD$',
    rf'thresh-low $\alpha = THRESHOLD_LOW$',
    'snz-p\n(p=95)',
    'snz-p\n(p=98)',
    'ISODATA',
    'trough-fill'
]

# make a bunch of boxplots
for scorename, data, medians in [('MCC', M, M_medians),
                                 ('precision', P, P_medians)]:

    # would like to combine these into the same plot
    # easier to do with xarray, maybe just do processing in a different script
    fig, ax = plt.subplots()
    boxplot_dict = ax.boxplot(data, labels=labels)
    axl = plt.setp(ax, xticklabels=labels)
    plt.setp(axl, rotation=90)
    ax.set_xlabel('segmentation method')
    ax.set_title(f'{scorename} scores of segmentation methods'
                 f'({quality_name} samples),'
                 f'{parametrization_name} parametrization')
    ax.set_ylabel(scorename)

    # label medians, from https://stackoverflow.com/a/18861734
    for line, med in zip(boxplot_dict['medians'], medians):
        x, y = line.get_xydata()[1] # right of median line
        plt.text(x, y, '%.2f' % med, verticalalignment='center')

    # you have to manually prevent clipping of rotated labels, amazing
    plt.subplots_adjust(bottom=0.30)
    plt.tight_layout()
    boxplot_name = '-'.join((quality_name, scorename, "boxplot",
                             parametrization_name))
    plt.savefig(os.path.join(OUTPUT_DIR, boxplot_name + '.png'))
    plt.show()
    plt.close()
```

APPENDIX B

3D VISUALIZATION OF THE FRANGI FILTER

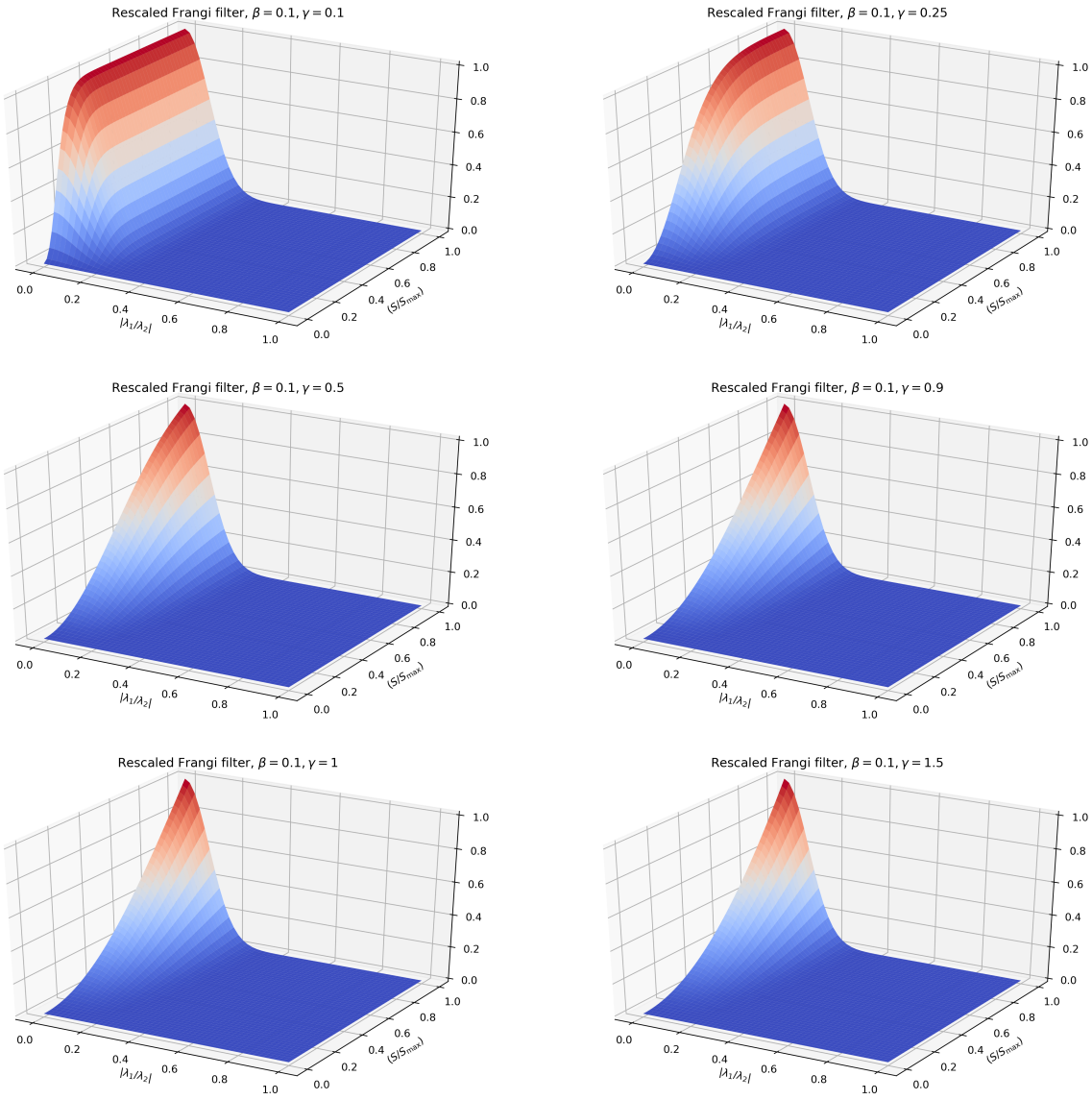


FIGURE 39: 3D graph of the Frangi Vesselness Measure, variable γ, $\beta = 0.1$

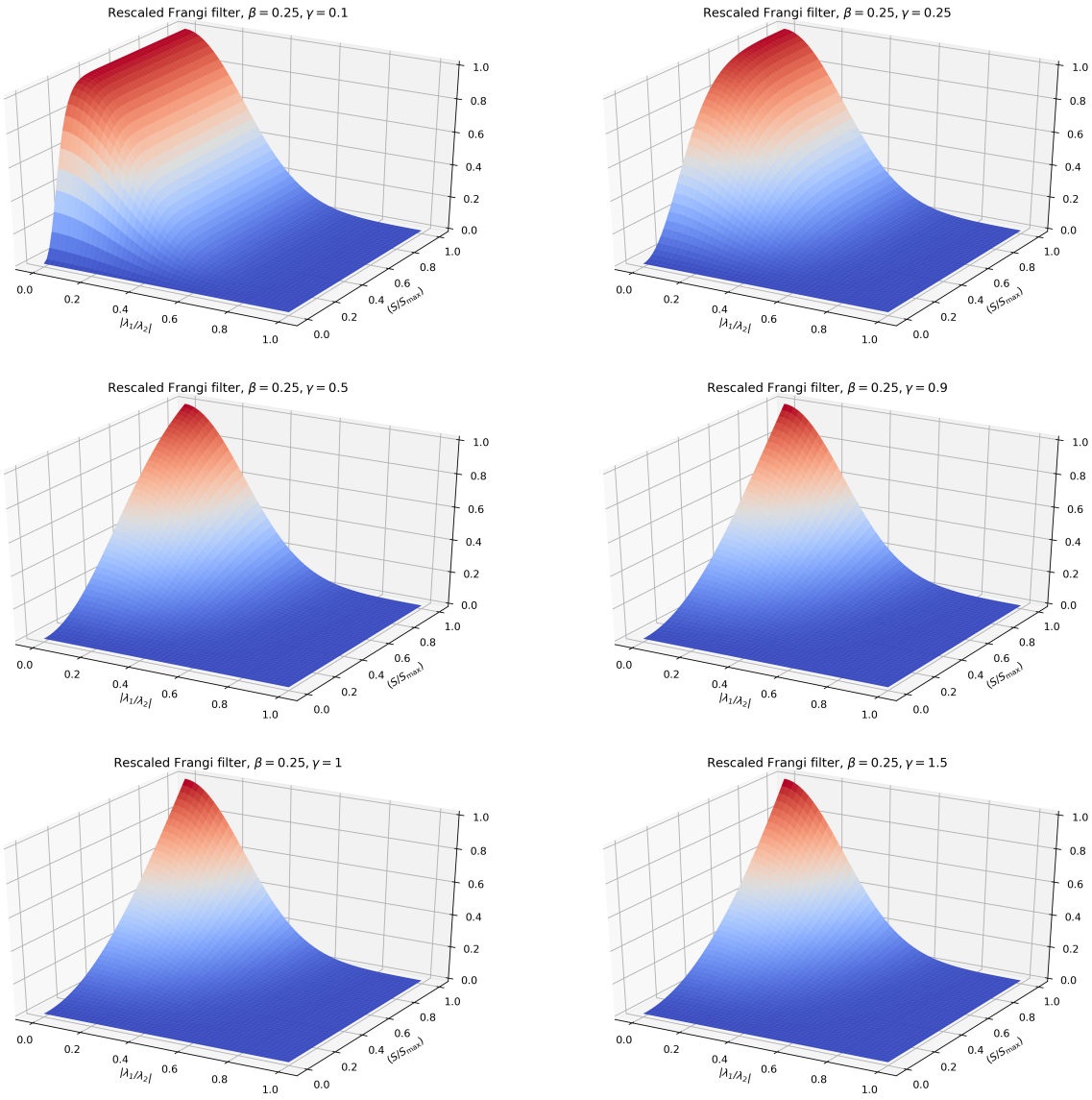


FIGURE 40: 3D graph of the Frangi Vesselness Measure, variable γ, $\beta = 0.25$

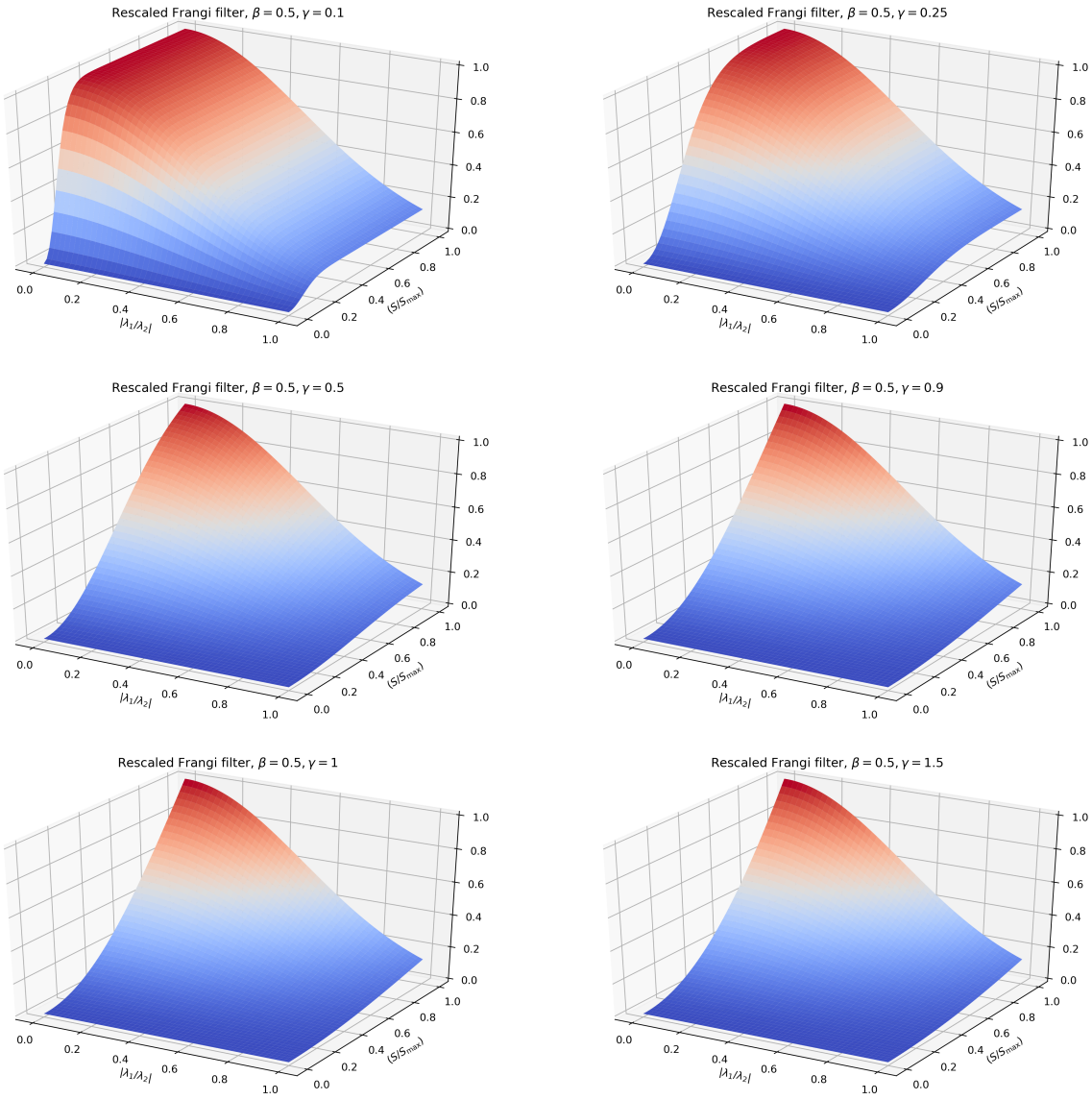


FIGURE 41: 3D graph of the Frangi Vesselness Measure, variable γ, $\beta = 0.5$

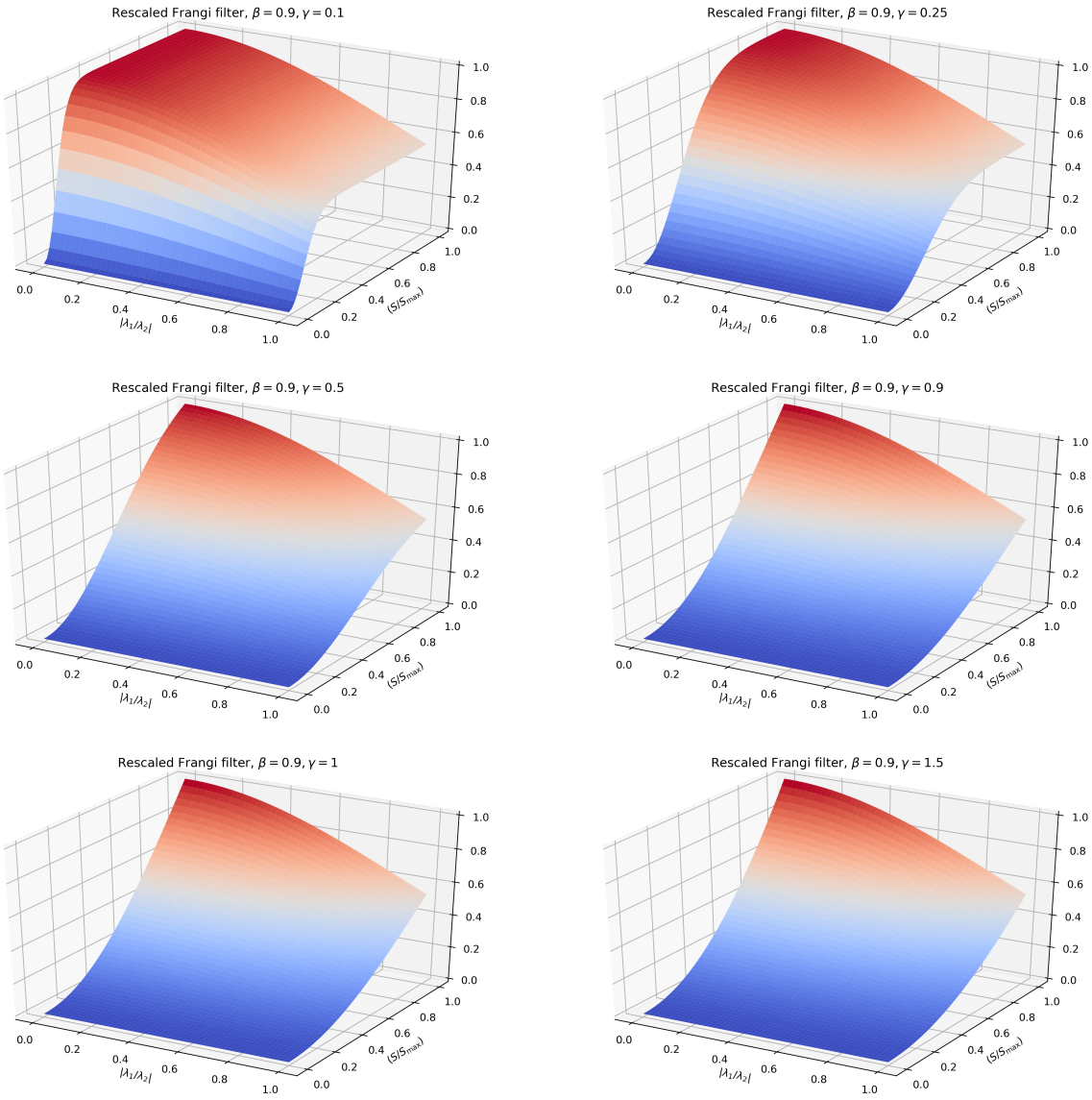


FIGURE 42: 3D graph of the Frangi Vesselness Measure, variable γ, $\beta = 0.9$

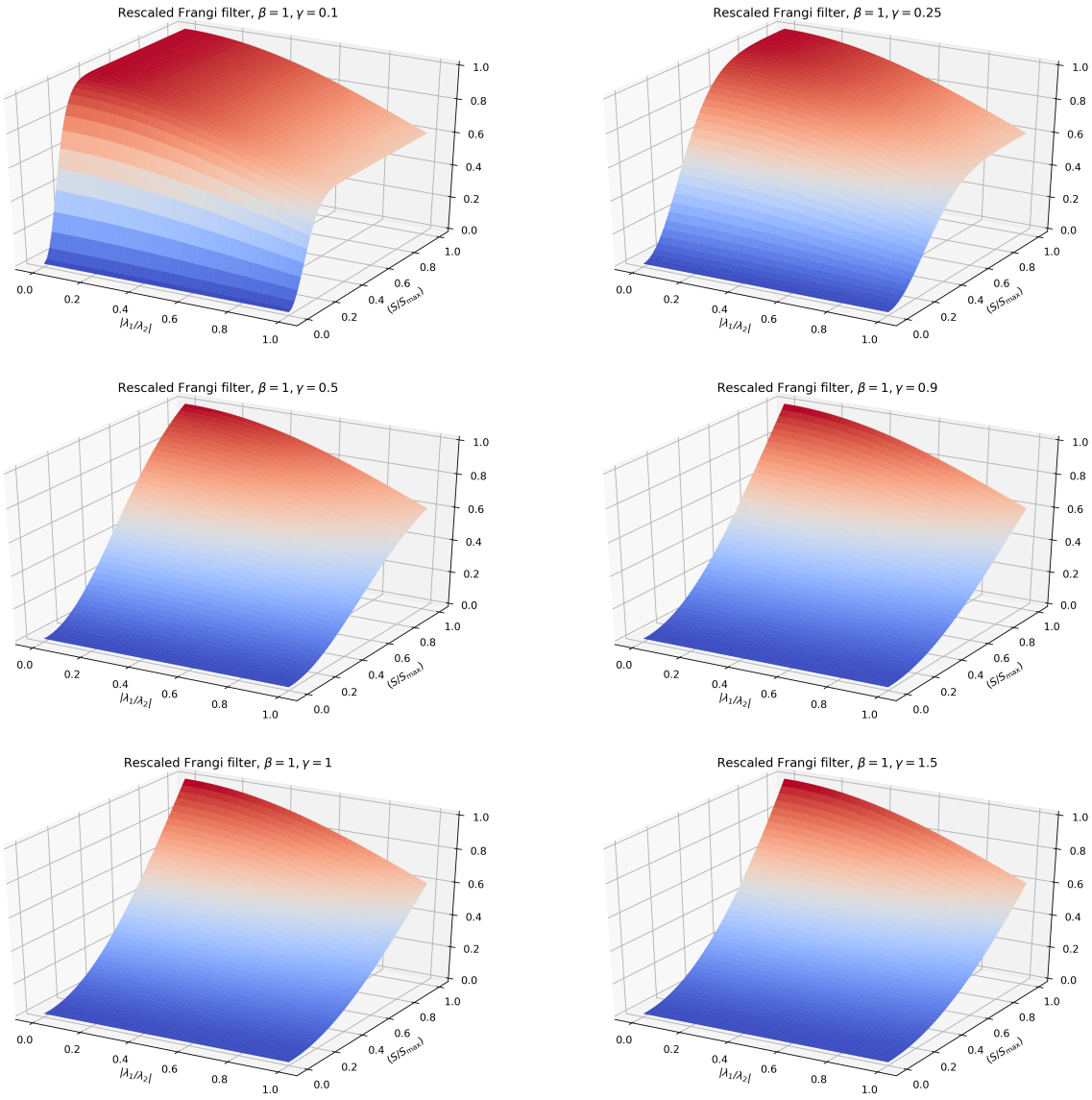


FIGURE 43: 3D graph of the Frangi Vesselness Measure, variable γ, $\beta = 1$

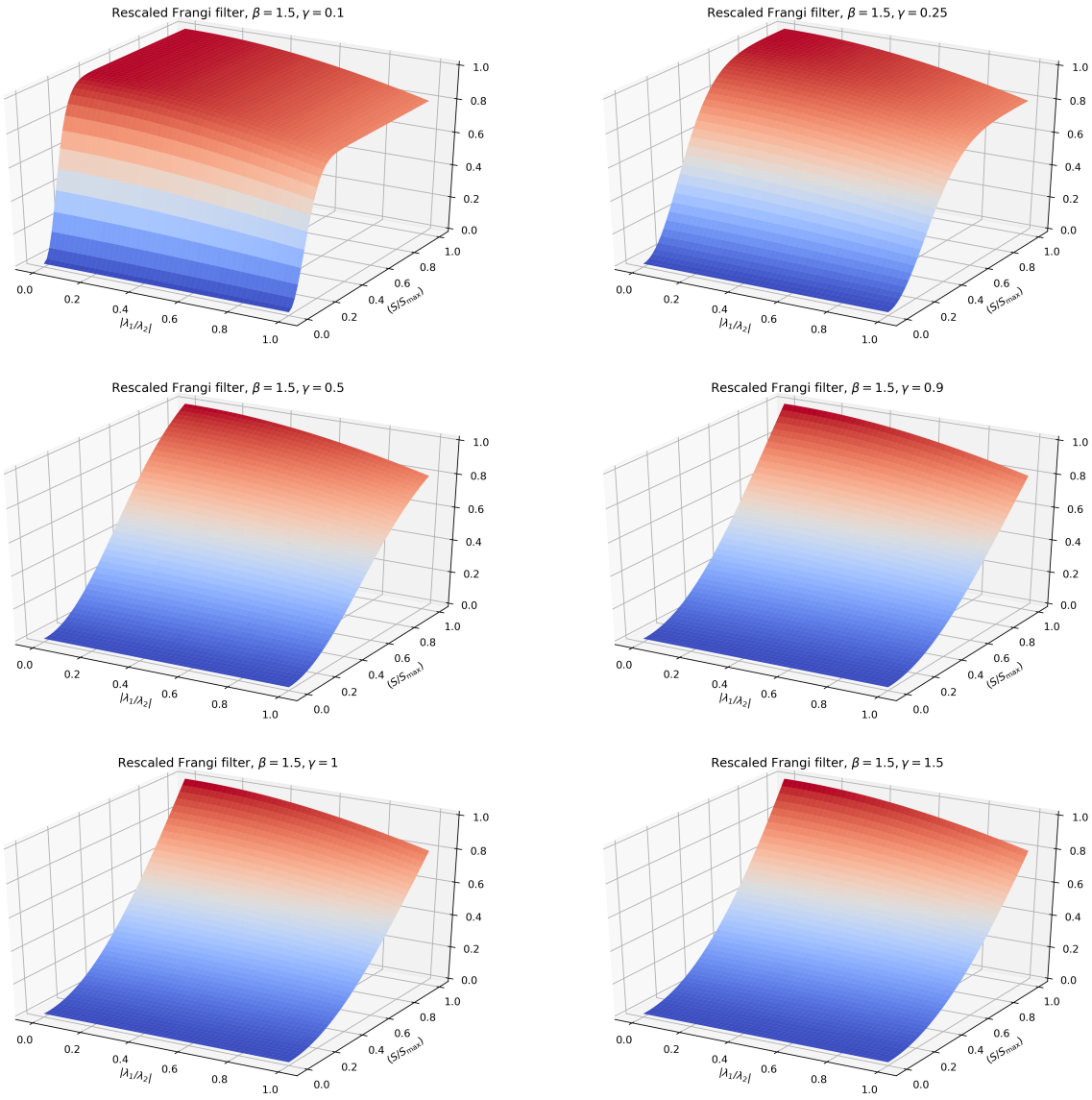


FIGURE 44: 3D graph of the Frangi Vesselness Measure, variable γ, $\beta = 1.5$

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] J.-M. Chang, H. Zeng, R. Han, Y.-M. Chang, R. Shah, C. M. Salafia, C. Newschaffer, R. K. Miller, P. Katzman, J. Moye, *et al.*, "Autism risk classification using placental chorionic surface vascular network features," *BMC medical informatics and decision making*, vol. 17, no. 1, p. 162, 2017.

[2] N. Huynh, *A filter bank approach to automate vessel extraction with applications*. PhD thesis, California State University, Long Beach, 2013.

[3] K. Y. Djima, C. Salafia, R. K. Miller, R. Wood, P. Katzman, C. Stodgell, and J.-M. Chang, "Enhancing placental chorionic surface vasculature from barium-perfused images with directional and multiscale methods," *Placenta*, vol. 57, pp. 292–293, 2017.

[4] Y.-M. Chang, R. Han, H. Zeng, R. Shah, C. Newschaffer, R. Miller, P. Katzman, J. Moye, C. Salafia, *et al.*, "Whole chorionic surface vessel feature analysis with the boruta method, and autism risk," *Placenta*, vol. 45, p. 75, 2016.

[5] N. Almoussa, B. Dutra, B. Lampe, P. Getreuer, T. Wittman, C. Salafia, and L. Vese, "Automated vasculature extraction from placenta images," in *Medical Imaging 2011: Image Processing*, vol. 7962, p. 79621L, International Society for Optics and Photonics, 2011.

[6] R. C. Gonzalez and R. E. Woods, "Digital image processing prentice hall," *Upper Saddle River, NJ*, 2002.

[7] W. Kühnel, B. Hunt, and A. M. Society, *Differential Geometry: Curves - Surfaces - Manifolds*. Student mathematical library, American Mathematical Society, 2006.

[8] J. H. Wilkinson, ed., *The Algebraic Eigenvalue Problem*. New York, NY, USA: Oxford University Press, Inc., 1988.

[9] R. Horn and C. Johnson, *Matrix Analysis*. Matrix Analysis, Cambridge University Press, 2012.

[10] X. Jiao and H. Zha, "Consistent computation of first-and second-order differential quantities for surface meshes," in *Proceedings of the 2008 ACM symposium on Solid and physical modeling*, pp. 159–170, ACM, 2008.

[11] R. Burden and J. Faires, *Numerical Analysis*. Brooks/Cole, 9 ed., 2011.

[12] A. F. Frangi, W. J. Niessen, K. L. Vincken, and M. A. Viergever, "Multiscale vessel enhancement filtering," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 130–137, Springer, 1998.

[13] Y. Sato, S. Nakajima, N. Shiraga, H. Atsumi, S. Yoshida, T. Koller, G. Gerig, and R. Kikinis, "Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images," *Medical image analysis*, vol. 2, no. 2, pp. 143–168, 1998.

[14] C. Lorenz, I. C. Carlsen, T. M. Buzug, C. Fassnacht, and J. Weese, "Multi-scale line segmentation with automatic estimation of width, contrast and tangential direction in 2d and 3d medical images," in *CVRMed-MRCAS'97* (J. Troccaz, E. Grimson, and R. Mösges, eds.), (Berlin, Heidelberg), pp. 233–242, Springer Berlin Heidelberg, 1997.

[15] S. D. Olabarriaga, M. Breeuwer, and W. Niessen, "Evaluation of hessian-based filters to enhance the axis of coronary arteries in ct images," in *International Congress Series*, vol. 1256, pp. 1191–1196, Elsevier, 2003.

[16] J. J. Koenderink, "The structure of images," *Biological Cybernetics*, vol. 50, pp. 363–370, Aug 1984.

[17] J. Sporring, *Gaussian Scale-Space Theory*. Norwell, MA, USA: Kluwer Academic Publishers, 1997.

[18] J. Babaud, M. Baudin, R. O. Duda, and A. P. Witkin, "Uniqueness of the gaussian kernel for scale-space filtering," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 8, pp. 26–33, 01 1986.

[19] T. Lindeberg, "Scale-space for discrete signals," *IEEE transactions on pattern analysis and machine intelligence*, vol. 12, no. 3, pp. 234–254, 1990.

[20] T. Lindeberg, *On the construction of a scale-space for discrete images*. KTH Royal Institute of Technology, 1988.

[21] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. New York: Dover, ninth dover printing, tenth gpo printing ed., 1964.

[22] T. Lindeberg, "Discrete derivative approximations with scale-space properties: A basis for low-level feature extraction," *Journal of Mathematical Imaging and Vision*, vol. 3, no. 4, pp. 349–376, 1993.

[23] T. Lindeberg, "Feature detection with automatic scale selection," *International journal of computer vision*, vol. 30, no. 2, pp. 79–116, 1998.

[24] B. Fornberg, "Generation of finite difference formulas on arbitrarily spaced grids," *Mathematics of computation*, vol. 51, no. 184, pp. 699–706, 1988.

[25] A. Morar, F. Moldoveanu, and E. Gröller, "Image segmentation based on active contours without edges," in *2012 IEEE 8th International Conference on Intelligent Computer Communication and Processing*, pp. 213–220, IEEE, 2012.

[26] E. Jones, T. Oliphant, P. Peterson, *et al.*, "SciPy: Open source scientific tools for Python," 2001–. [Online; accessed <today>].

[27] S. Damelin and N. Hoang, "On surface completion and image inpainting by biharmonic functions: Numerical aspects," *International Journal of Mathematics and Mathematical Sciences*, vol. 2018, 2018.

[28] H. Lange, "Automatic glare removal in reflectance imagery of the uterine cervix," in *Medical Imaging 2005: Image Processing*, vol. 5747, pp. 2183–2193, International Society for Optics and Photonics, 2005.

[29] B. Matthews, "Comparison of the predicted and observed secondary structure of t4 phage lysozyme," *Biochimica et Biophysica Acta (BBA) - Protein Structure*, vol. 405, no. 2, pp. 442–451, 1975.

[30] L. Grady, "Random walks for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, pp. 1768–1783, Nov. 2006.

[31] C. Anghel, K. Archer, J.-M. Chang, A. Cochran, A. Radulescu, C. M. Salafia, R. Turner, K. Y. Djima, and L. Zhong, "Placental vessel extraction with shearlets, laplacian eigenmaps, and a conditional generative adversarial network," in *Understanding Complex Biological Systems with Mathematics*, pp. 171–196, Springer, 2018.

[32] T. Ridler, S. Calvard, *et al.*, "Picture thresholding using an iterative selection method," *IEEE trans syst Man Cybern*, vol. 8, no. 8, pp. 630–632, 1978.

[33] Z. Guo and R. W. Hall, "Parallel thinning with two-subiteration algorithms," *Communications of the ACM*, vol. 32, no. 3, pp. 359–373, 1989.