

ABSTRACT

A Beefy Frangi Filter for Noisy Vascular Segmentation and Network Connection in PCSVN

By

Lucas Wukmer

May 2018

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. 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 health on later conditions, but may ideally serve as a real-time diagnostic for neonatal risk factors as well.

Much work has been to develop reliable vascular extraction 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 arguments to placental images, but this approach is greatly hindered by the inherent irregularity of the placental surface as a whole, which introduces significant noise into the image domain. A recent attempt was made to apply an additional local curvilinear filter to the Frangi result in an effort to remove some noise from the final extraction.

Here we propose an alternate extraction method. First, we use arguments from Frangi's original paper to provide a proper selection of parameters for our particular image domain. Using the same arguments from differential geometry that gave rise to the Frangi filter, we calculate the leading principal direction (eigenvector of the Hessian) to indicate the directionality of curvilinear features at a particular scale. We are then able to apply an appropriately-oriented morphological filter to our Frangi targets at select scales to remove noise. This approach differs significantly from previous efforts in that morphological filtering will take place at each scale space, rather than

being performed one time following multiscale synthesis. Noise removal performed in this way is expected to aide in coherent interpretation of targets that should appear in a connected network.

Finally, we discuss an important advancement in implementation-scale space conversion for differentiation (i.e. gaussian blur) via Fast Fourier Transform (FFT) rather than a more traditional convolution with a gaussian kernel, which offers a significant speedup. This thesis will also contain a general, in depth summary of both multiscale Hessian filters and scale-space theory.

We demonstrate the effectiveness of our improved vascular extraction technique on several of the following image domains: a private database of barium-injected samples provided by University of Rochester, uninjected/raw placental samples from Placental Analytics LLC, a collection of simulated images, the DRIVE and STARE databases of retinal MRAs, and a new collection of computer-generated images with significant curvilinear content.

Time permitting, this research will be extended to include a method of network connection, so that a logically connected vascular network is realized (i.e. network completion).

A Beefy Frangi Filter for Noisy Vascular Segmentation and Network Connection in PCSVN

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 Wukmer

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

May 2018

ACKNOWLEDGEMENTS

Acknowledgments go here.

TABLE OF CONTENTS

ABSTRACT	ii
ACKNOWLEDGEMENTS	ii
LIST OF TABLES	iv
LIST OF FIGURES	v
1. INTRODUCTION	1
2. MATHEMATICAL METHODS	2
Linear-shift and Rotational invariance	26
Continuity of scale parameter	26
Causality condition	27
Fourier Transform of a continuous 1D signal.....	33
Fourier Transform of a Discrete 1D signal	34
1D Discrete Fourier Transform (DFT)	34
2D Discrete Fourier Transform Convolution Theorem.....	35
FFT	37
3. IMPLEMENTATIONS	42
4. RESEARCH PROTOCOL	43
5. RESULTS AND ANALYSIS	50
6. CONCLUSION	51
7. FUTURE RESEARCH DIRECTIONS	52
APPENDICES	53
A. APPENDIX TITLE	54
A APPENDIX TITLE	54
BIBLIOGRAPHY	56

LIST OF TABLES

LIST OF FIGURES

1	Tangent plane of a graph.....	6
2	The graph of a cylindrical ridge of radius r , as given by $f(x,y)$ as given above.	17
3	The principal eigenvectors at a ridge like structure	22
4	A representative placental NCS sample with vascular tracing	43
5	Preprocessed files from an NCS sample	44
6	Demonstration of boundary dilation	46
7	Frangi vesselness score at several scales.....	48
8	Frangi vesselness score at several scales (inset).....	49

CHAPTER 1

INTRODUCTION

The Applied Problem

Reference Nen's paper and latest autism risk paper.

Context in Image Processing

- brief background of math image processing methods
- what's been tried in this applied problem
 - nen [4]
 - catalina's paper
 - kara's paper
 - other domains

Research Goals

Segue from previous paragraph, talk about strengths and weaknesses of other methods and what this research aims to accomplish. Include ‘research questions’ that could allow a reader to answer the question ”will this research work for my problem?”. ”Elevator pitch” maybe goes here.

Roadmap

Outline of the thesis (“firstly” bullshit)

CHAPTER 2

MATHEMATICAL METHODS

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 suitable 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.

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).

$$\mathbf{I} \in \mathbb{N}^{M \times N} \quad \text{with} \quad 0 \leq I_{ij} \leq 2^8 - 1$$

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).

$$h : \mathbb{R}^2 \rightarrow \mathbb{R} \quad \text{with} \quad h \in C^2(\mathbb{R}^2), \quad \text{where} \quad h(i, j) = I_{ij} \quad \forall (i, j) \in \{0, \dots, M\} \times \{0, \dots, N\} \subset \mathbb{N}^2$$

That is, the function h is identical to the pixel matrix \mathbf{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 \mathbf{I} is not really a perfect representation of the underlying “content” within the picture. Not only is information lost when \mathbf{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 [2], especially when the goal is noise reduction. However, we will be content to simply represent the pixels of \mathbf{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), 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 \mathbf{I} as a sampling again to motivate our development of scale space theory in section 2.4, we ultimately use these techniques because we find them successful to our problem.

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, following [7] (although any undergraduate text in Differential Geometry should prove satisfactory).

Preliminaries of Differential Geometry

Given an open subset $U \subset \mathbb{R}^2$ and a twice differentiable function $h : U \rightarrow \mathbb{R}$ (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*

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

Since the graph f is clearly one-to-one by definition, we may readily associate any input $u \in U$ with its corresponding output $p \in f[U]$, i.e. $p = f(u) = f(u_1, u_2) = (u_1, u_2, h(u_1, u_2))$, 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 so called “differential” of f , the lattermost of which we

will only define for the immediate case required. Seeing that f is one-to-one should make a lot of this futzing about complete overkill, but I've yet to find a way to distill it. That is, this development works for any parametrized surface element, not necessarily a graph. Whatever for now.

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

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

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

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

Although not necessary presently, we could just as easily consider the differential of an arbitrary function as a map between tangent vectors in the function's domain and tangent vectors in its range. We could also just identify this as mapping $U \rightarrow \mathbb{R}^3$ by the obvious isomorphism described above. and then differential of f at x is simply a linear transformation of between the tangent spaces $T_u U$ and $T_p \mathbb{R}^3$ where the transformation in question 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).

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

This vectors of 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, $\text{span} \left\{ \frac{\partial f}{\partial u_1}, \frac{\partial f}{\partial u_2} \right\} = T_u f$.

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 $T_u f$.*

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.

Proof. Given the definition of a graph f as in definition 2.3, we can directly calculate the partial 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. \square

The partials 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.

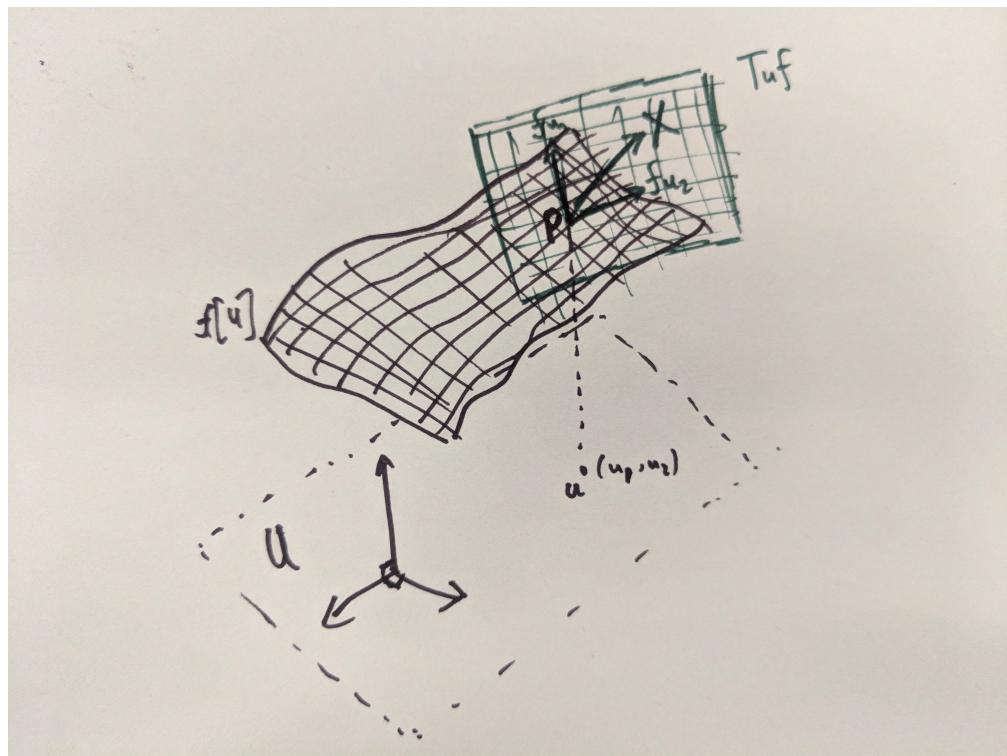


FIGURE 1: Tangent plane of a graph

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 .

Curvature of a surface and its calculation

In the context of a regular arc-length parametrized curve $c : I \rightarrow \mathbb{R}^3$ parametrized along some closed interval $I \subset \mathbb{R}$ (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 $c : I \rightarrow \mathbb{R}^3$ 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_c so that $c = f \circ \theta_c$. That is,*

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

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:

$$-\frac{dc}{dt} = -\frac{d}{dt}[f(\theta_c(t))] \tag{2.1}$$

$$= -\frac{d}{dt}[f(\theta_1(t), \theta_2(t))] \tag{2.2}$$

$$= \theta'_1(t) \left(\frac{\partial f}{\partial u_1} \right) + \theta'_2(t) \left(\frac{\partial f}{\partial u_2} \right) \in T_u f. \tag{2.3}$$

Considering a point $p \in I$ and its associated point $u = \theta_c(p)$, we wish to compare the curvatures of all (regular) surface curves passing through the point p 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 $X \in T_{uf}$, any regular curve on the surface $c : I \rightarrow \text{image}(f)$ 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'll require one final definition.

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

$$v : U \rightarrow \mathbb{R}^3 \quad \text{by} \quad v(u) := \frac{\frac{\partial f}{\partial u_1} \times \frac{\partial f}{\partial u_2}}{\left\| \frac{\partial f}{\partial u_1} \times \frac{\partial f}{\partial u_2} \right\|}$$

Each partial above understood to be evaluated at the input $u \in U$; that is, we calculate $\left. \frac{\partial f}{\partial u_i} \right|_u$. The existence of the cross product in its definition makes it clear that $v \perp \frac{\partial f}{\partial u_i}$ each $i = 1, 2$. A simple dimensionality argument of \mathbb{R}^3 implies that these must exist in T_{uf} . However, we can also show it directly:

To show that $\left\{ \frac{\partial v}{\partial u_1}, \frac{\partial v}{\partial u_2} \right\} \in T_{uf}$, first note that at any particular $u \in U$, $\langle v, v \rangle = 1 \implies \frac{\partial}{\partial u_i} \langle v, v \rangle = 0$, and so by chain rule $2 \langle \frac{\partial v}{\partial u_i}, v \rangle = 0 \implies \frac{\partial v}{\partial u_i} \perp v$. Since $v \perp \text{span} \left\{ \frac{\partial f}{\partial u_i} \right\}$ as well (since v its outer product), in \mathbb{R}^3 , this implies $\text{span} \left\{ \frac{\partial v}{\partial u_i} \right\} \parallel \text{span} \left\{ \frac{\partial f}{\partial u_i} \right\}$.

Thus, we have $\text{span} \left\{ \frac{\partial v}{\partial u_1}, \frac{\partial v}{\partial u_2} \right\} \subset T_{uf}$ as well and we can also use it as a basis.

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

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

$$c'' = \langle c'', X \rangle X + \langle c'', v \rangle v \tag{2.4}$$

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. (2.4) using the chain rule:

$$\langle c'', v \rangle = \frac{\partial}{\partial t} [\langle c', v \rangle] - \langle c', \frac{\partial v}{\partial t} \rangle \quad (2.5)$$

$$= \frac{\partial}{\partial t} [\langle X, v \rangle] - \langle c', \frac{\partial v}{\partial t} \rangle \quad (2.6)$$

$$= 0 - \langle X, \frac{\partial v}{\partial t} \rangle \quad (2.7)$$

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

$$\|c''\| = \|\langle c'', X \rangle X + \langle c'', v \rangle v\| = \|0 + \langle c'', v \rangle v\| \quad (2.8)$$

$$= -\langle X, \frac{\partial v}{\partial t} \rangle \|v\| \quad (2.9)$$

$$= -\langle X, \frac{\partial v}{\partial t} \rangle \quad (2.10)$$

$$= \langle X, -\frac{\partial v}{\partial t} \rangle \quad (2.11)$$

We may compute the quantity $-\frac{\partial v}{\partial t}$ that appears in eq. (2.11) via chain rule:

$$-\frac{dv}{dt} = -\frac{d}{dt} [v(u_1, u_2)] \quad (2.12)$$

$$= -\frac{d}{dt} [v(\theta_1(t), \theta_2(t))] \quad (2.13)$$

$$= \theta'_1(t) \left(-\frac{\partial v}{\partial u_1} \right) + \theta'_2(t) \left(-\frac{\partial v}{\partial u_2} \right) \quad (2.14)$$

Identifying $\text{span} \left\{ -\frac{\partial v}{\partial u_i} \right\}_{i=1,2}$ as a subset of $T_u f$, we can define a linear transformation L which maps the basis $\left\{ \frac{\partial f}{\partial u_i} \right\}_{i=1,2}$ to this subset:

Definition 2.10 (The Weingarten Map).

$$L : T_u f \rightarrow T_u f \quad \text{given by the composition} \quad L = Dv \circ (Df)^{-1}.$$

That is, $L(\frac{\partial f}{\partial u_i}) = -\frac{\partial v}{\partial u_i}$ for $i = 1, 2$, where the negative sign comes about from blind adherence to eq. (2.14) and eq. (2.11). This allows us to rewrite the time derivative of the Gauss map eq. (2.12) as

$$-\frac{dv}{dt} = \theta'_1(t) \left(-\frac{\partial v}{\partial u_1} \right) + \theta'_2(t) \left(-\frac{\partial v}{\partial u_2} \right) \quad (2.15)$$

$$= \theta'_1(t) \left(L \left(\frac{\partial f}{\partial u_1} \right) \right) + \theta'_2(t) \left(L \left(\frac{\partial f}{\partial u_2} \right) \right) \quad (2.16)$$

$$= L \left[\theta'_1(t) \left(\frac{\partial f}{\partial u_1} \right) + \theta'_2(t) \left(\frac{\partial f}{\partial u_2} \right) \right] \quad (2.17)$$

$$= L \left(\frac{d}{dt} [f(\theta(t))] \right) = L \left(\frac{d}{dt} [c(t)] \right) = L(X) \quad (2.18)$$

With this, we can re-express the curvature of our curve from eq. (2.11) as the much simpler

$$\|c''\| = \langle X, -\frac{\partial v}{\partial t} \rangle = \langle X, L(X) \rangle \quad (2.19)$$

The linear transformation L from definition 2.10, and thereby the computation of curvature given in eq. (2.19), depends only on the point u and the selected direction X , not on the particular curve c at all. \square

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_v at point u in the direction X is given by*

$$\kappa_v := \langle X, L(X) \rangle$$

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 $L(\frac{\partial f}{\partial u_i}) = -\frac{\partial v}{\partial u_i}$.

That is, we may trace any $X \in T_{uf}$ 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 span of $\left\{ -\frac{\partial v}{\partial u_1}, -\frac{\partial v}{\partial u_2} \right\}$.

The Weingarten map can be formally shown to be well-defined, invariant under coordinate transformation [7] in the general case, which is certainly useful for surfaces f that are not graphs. The situation is much less delicate if f is a graph—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 $\widehat{L} \in R^{2 \times 2}$) we simply wish to find a linear transformation such that $\widehat{L} \frac{\partial f}{\partial u_i} \Big|_{T_{uf}} = -\frac{\partial v}{\partial u_i} \Big|_{T_{uf}}$ for $i = 1, 2$ where $-X|_{T_{uf}}$ denotes that $X \in T_{uf}$ is being represented in so-called 'local coordinates' for T_{uf} (Strictly speaking, of course $T_{uf} \subset \mathbb{R}^3$ and thus $\frac{\partial f}{\partial u_i} \in \mathbb{R}^3$. Thus when we say $\frac{\partial f}{\partial u_i} \Big|_{T_{uf}}$ we are referring to this 3-vector expanded with respect to a basis for T_{uf}). In matrix form, we describe this situation as

$$\begin{bmatrix} \widehat{L} \end{bmatrix} \begin{bmatrix} \frac{\partial f}{\partial u_1} \Big|_{T_{uf}} & \frac{\partial f}{\partial u_2} \Big|_{T_{uf}} \\ \downarrow & \downarrow \end{bmatrix} = \begin{bmatrix} \widehat{L} \frac{\partial f}{\partial u_1} \Big|_{T_{uf}} & \widehat{L} \frac{\partial f}{\partial u_2} \Big|_{T_{uf}} \\ \downarrow & \downarrow \end{bmatrix} \quad (2.20)$$

$$= \begin{bmatrix} \frac{\partial v}{\partial u_1} \Big|_{T_{uf}} & -\frac{\partial v}{\partial u_2} \Big|_{T_{uf}} \\ \downarrow & \downarrow \end{bmatrix} \quad (2.21)$$

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

$$\Rightarrow \begin{bmatrix} \hat{L} \\ \end{bmatrix} \begin{bmatrix} \leftarrow \frac{\partial f}{\partial u_1} \rightarrow \\ \leftarrow \frac{\partial f}{\partial u_2} \rightarrow \\ \downarrow \\ \end{bmatrix} \begin{bmatrix} \uparrow & \uparrow \\ \frac{\partial f}{\partial u_1} & \frac{\partial f}{\partial u_2} \\ \downarrow & \downarrow \\ \end{bmatrix} = \begin{bmatrix} \leftarrow \frac{\partial f}{\partial u_1} \rightarrow \\ \leftarrow \frac{\partial f}{\partial u_2} \rightarrow \\ \downarrow \\ \end{bmatrix} \begin{bmatrix} \uparrow & \uparrow \\ -\frac{\partial v}{\partial u_1} & -\frac{\partial v}{\partial u_2} \\ \downarrow & \downarrow \\ \end{bmatrix} \quad (2.22)$$

We can simplify this greatly by defining

$$g_{ij} := \langle \frac{\partial f}{\partial u_i}, \frac{\partial f}{\partial u_j} \rangle \quad \text{and} \quad h_{ij} := \langle \frac{\partial f}{\partial u_i}, -\frac{\partial v}{\partial u_j} \rangle \quad (2.23)$$

so that

$$\begin{bmatrix} \hat{L} \\ \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \\ \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \\ \end{bmatrix} \quad (2.24)$$

Then we rearrange to solve for \hat{L} as

$$\hat{L} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \\ \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \\ \end{bmatrix}^{-1} \quad (2.25)$$

where $[g_{ij}]$ 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 $f : U \rightarrow \mathbb{R}^3$ 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.

Principal Curvatures and Principal Directions

To do so, we shall consider the relationship between the direction X and the normal curvature κ_v 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 (the ‘ith’) component of $\nabla_v \langle v, Av \rangle$:

$$\left(\nabla_v \langle v, Av \rangle \right)_i = \frac{\partial}{\partial v_i} \left[\langle v, Av \rangle \right] = \frac{\partial}{\partial v_i} \left[\sum_{j=1}^n v_j (Av)_j \right] \quad (2.26)$$

$$= \frac{\partial}{\partial v_i} \left[\sum_{j=1}^n v_j \sum_{k=1}^n a_{jk} v_k \right] \quad (2.27)$$

$$= \frac{\partial}{\partial v_i} \left[a_{ii} v_i^2 + v_i \sum_{k \neq i} a_{ik} v_k + v_i \sum_{j \neq i} a_{ji} v_j + \sum_{j \neq i} \sum_{k \neq i} v_j a_{jk} v_k \right] \quad (2.28)$$

$$= 2a_{ii}v_i + \sum_{k \neq i} a_{ik}v_k + \sum_{j \neq i} a_{ji}v_j + 0 \quad (2.29)$$

$$= 2a_{ii}v_i + 2 \sum_{k \neq i} a_{ik}v_k = 2 \sum_{k=1}^n a_{ik}v_k = 2(Av)_i \quad (2.30)$$

$$\implies \nabla_v \langle v, Av \rangle = 2Av. \quad (2.31)$$

□

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 $\kappa_v = \langle LX, X \rangle$ 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{L} . Using the method of Lagrange multipliers, we define the Lagrangian:

$$\mathcal{L}(X; \lambda) := \langle \widehat{L}X, X \rangle - \lambda(\langle X, X \rangle - 1) \quad (2.32)$$

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

$$\begin{cases} \nabla_X \langle \hat{L}X, X \rangle - \lambda \nabla_X (\langle X, X \rangle - 1) = 0 \\ \langle X, X \rangle - 1 = 0 \end{cases} \quad (2.33)$$

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

$$\begin{aligned} \nabla_X \langle \hat{L}X, X \rangle - \lambda \nabla_X (\langle X, X \rangle - 1) &= 0 \\ 2\hat{L}X - \lambda(2X) &= 0 \\ \implies \hat{L}X - \lambda X &= 0 \\ \implies \hat{L}X &= \lambda X \end{aligned} \quad (2.34)$$

which implies that X is an eigenvector of \hat{L} with corresponding eigenvalue λ ($X \neq 0$ from the second equation of eq. (2.33)). 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. \square

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. When $f : U \rightarrow \mathbb{R}^3$ is given by $(x, y) \mapsto (x, y, h(x, y))$, the matrix representation of the Weingarten map is exactly the Hessian matrix given in (2.1) is given by

$$\widehat{\mathbf{L}} = \text{Hess}(h)\tilde{G}, \quad \text{where } \tilde{G} := \frac{1}{\sqrt{1+h_x^2+h_y^2}} \begin{bmatrix} 1+h_y^2 & -h_x h_y \\ -h_x h_y & 1+h_x^2 \end{bmatrix} \quad (2.35)$$

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

Proof. First, we can (using chain rule) rewrite each component as in eq. (2.23):

$$h_{ij} = \left\langle \frac{\partial f}{\partial u_i}, -\frac{\partial \mathbf{v}}{\partial u_j} \right\rangle = \left\langle \frac{\partial^2 f}{\partial u_i \partial u_j}, \mathbf{v} \right\rangle$$

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

$$\begin{aligned} f_x &= (1, 0, h_x), & f_y &= (0, 1, h_y) \\ f_{xx} &= (0, 0, h_{xx}), & f_{xy} &= (0, 0, h_{xy}) = f_{yx}, & f_{yy} &= (0, 0, h_{yy}) \end{aligned} \quad (2.36)$$

and we have the unit normal vector (Gauss map)

$$\mathbf{v}(u_1, u_2) = \frac{\frac{\partial f}{\partial x} \times \frac{\partial f}{\partial y}}{\left\| \frac{\partial f}{\partial x} \times \frac{\partial f}{\partial y} \right\|} \quad (2.37)$$

$$= \frac{(1, 0, h_x) \times (0, 1, h_y)}{\|\dots\|} \quad (2.38)$$

$$= \frac{(-h_x, -h_y, 1)}{\sqrt{h_x^2 + h_y^2 + 1}} \quad (2.39)$$

We then calculate each h_{ij} as

$$\begin{aligned} h_{11} &= \left\langle \frac{\partial^2 f}{\partial x^2}, \mathbf{v} \right\rangle = \frac{h_{xx}}{\sqrt{1+h_x^2+h_y^2}} \\ h_{12} &= \left\langle \frac{\partial^2 f}{\partial x \partial y}, \mathbf{v} \right\rangle = \frac{h_{xy}}{\sqrt{1+h_x^2+h_y^2}} = h_{21} \\ h_{22} &= \left\langle \frac{\partial^2 f}{\partial y^2}, \mathbf{v} \right\rangle = \frac{h_{yy}}{\sqrt{1+h_x^2+h_y^2}} \end{aligned} \quad (2.40)$$

and thus the first matrix in eq. (2.25) is given by

$$[h_{ij}] = \frac{1}{\sqrt{1+h_x^2+h_y^2}} \text{Hess}(h) \quad (2.41)$$

To calculate the second, we use

$$\begin{aligned} g_{ij} &= \left\langle \frac{\partial f}{\partial u_i}, \frac{\partial f}{\partial u_j} \right\rangle \\ g_{11} &= \langle f_x, f_x \rangle = 1 + h_x^2 \\ g_{12} &= \langle f_x, f_y \rangle = h_x h_y = g_{21} \\ g_{22} &= \langle f_y, f_y \rangle = 1 + h_y^2 \end{aligned} \quad (2.42)$$

and thus

$$[g_{ij}]^{-1} = \begin{bmatrix} 1 + h_x^2 & h_x h_y \\ h_x h_y & 1 + h_y^2 \end{bmatrix}^{-1} = \begin{bmatrix} 1 + h_y^2 & -h_x h_y \\ -h_x h_y & 1 + h_x^2 \end{bmatrix} \quad (2.43)$$

Combining $[h_{ij}]$ and $[g_{ij}]^{-1}$ from eq. (2.43) and eq. (2.41) we arrive at eq. (2.35). \square

Thus the matrix of the Weingarten map \widehat{L} is the Hessian matrix exactly at a critical point $u \in U$, where $\nabla h(u) = (h_x(u), h_y(u)) = 0$. Of course this implies that \widehat{L} and $\text{Hess}(h)$ have the same eigenvalues and eigenvectors at these points.

To make this a little more explicit, we will calculate the Weingarten map for a relatively simple graph.

The Weingarten map and Principal Curvatures of a Cylindrical Ridge

Let f be the graph given by

$$f : \mathbb{R}^2 \rightarrow \mathbb{R}^3 \text{ by } f(x, y) = (x, y, h(x, y)), \text{ with } h(x, y) = \begin{cases} \sqrt{r^2 - x^2} & -r \leq x \leq r \\ 0 & \text{else} \end{cases} \quad (2.44)$$

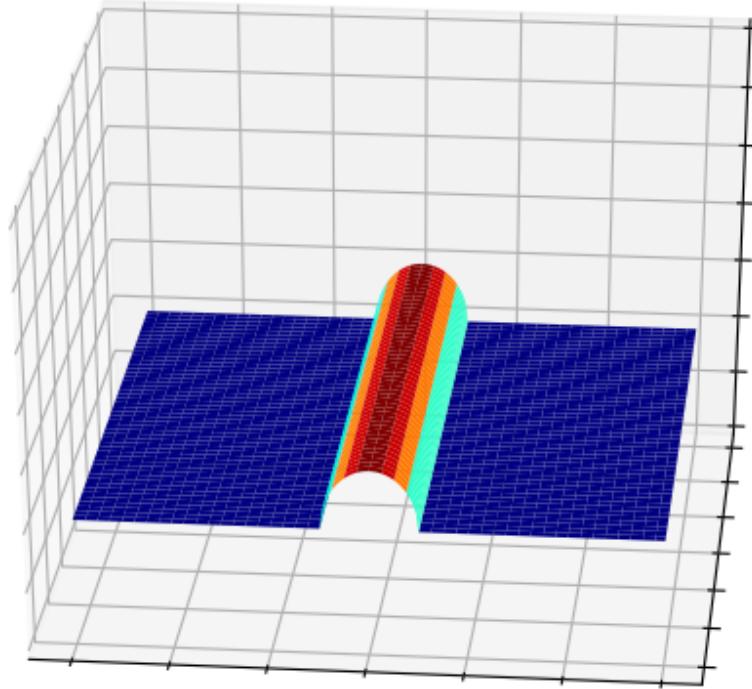


FIGURE 2: The graph of a cylindrical ridge of radius r , as given by $f(x,y)$ as given above.

We calculate the necessary partial derivatives of f as follows:

$$\frac{\partial f}{\partial x} = \left(1, 0, \frac{-x}{\sqrt{r^2 - x^2}} \right) , \quad \frac{\partial^2 f}{\partial x^2} = \left(0, 0, \frac{-r^2}{(\sqrt{r^2 - x^2})^3} \right) \quad (2.45)$$

$$\frac{\partial f}{\partial y} = (0, 1, 0) , \quad \frac{\partial^2 f}{\partial y^2} = \frac{\partial^2 f}{\partial x \partial y} = 0 \quad (2.46)$$

The gauss map is given by

$$\mathbf{v}(x,y) = \frac{\frac{\partial f}{\partial x} \times \frac{\partial f}{\partial y}}{\left\| \frac{\partial f}{\partial x} \times \frac{\partial f}{\partial y} \right\|} = \left(\frac{x}{r}, 0, \frac{\sqrt{r^2 - x^2}}{r} \right) \quad (2.47)$$

$$\implies \frac{\partial \mathbf{v}}{\partial x} = \left(\frac{1}{r}, 0, \frac{-x}{r\sqrt{r^2 - x^2}} \right), \quad \frac{\partial \mathbf{v}}{\partial y} = (0, 0, 0). \quad (2.48)$$

We then calculate matrix elements of the Weingarten map's construction as given in eq. (2.41) and eq. (2.43) :

$$[h_{ij}] = \frac{1}{\sqrt{1+h_x^2+h_y^2}} \text{Hess}(h) = \frac{1}{\sqrt{1+\left(\frac{x^2}{r^2-x^2}\right)}} \begin{bmatrix} \frac{-r^2}{\sqrt{r^2-x^2}} & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \frac{-r}{r^2-x^2} & 0 \\ 0 & 0 \end{bmatrix} \quad (2.49)$$

$$[g_{ij}]^{-1} = \begin{bmatrix} \frac{r^2-x^2}{r^2} & 0 \\ 0 & 1 \end{bmatrix} \quad (2.50)$$

$$\implies \widehat{\mathbf{L}} = [h_{ij}][g_{ij}]^{-1} = \begin{bmatrix} \frac{-r}{r^2-x^2} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{r^2-x^2}{r^2} & 0 \\ 0 & 1 \end{bmatrix} \quad (2.51)$$

$$= \begin{bmatrix} -\frac{1}{r} & 0 \\ 0 & 0 \end{bmatrix} \quad (2.52)$$

We see that $u_2 = (0, 1)$ and $u_1 = (1, 0)$ are eigenvectors for $\widehat{\mathbf{L}}$ with respective eigenvalues $\kappa_2 = -\frac{1}{r}$, $\kappa_1 = 0$. Given the theorem of Olinde Rodriguez 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 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 (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.

Viewing the surface in \mathbb{R}^3 , we define the Hessian Hess of the surface L at a point (x, y) on the

surface as the matrix of its second partial derivatives:

$$\text{Hess}(x,y) = \begin{bmatrix} L_{xx}(x,y) & L_{xy}(x,y) \\ L_{yx}(x,y) & L_{yy}(x,y) \end{bmatrix} \quad (2.53)$$

At any point (x,y) we denote the two eigenpairs of $\text{Hess}(x,y)$ as

$$\text{Hess}u_i = \kappa_i u_i, \quad i = 1, 2 \quad (2.54)$$

where κ_i and u_i are known as the *principal curvatures* and *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 lemma:

Lemma 2.7 (Principal Axis Theorem?). *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

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

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 $v_1, v_2 \neq 0$. ¹

¹To simplify notation, we simplify our argument to consider two explicit eigenvectors only, since we're only concerned with the 2×2 matrix Hess anyway.

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 $L(x,y)$.

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

The Frangi Filter

The Frangi filter, first described by Alejandro Frangi et al. in [1] is a widely used (cite) 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 [10] and [8], as well as a comparison given in [9]. 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 originally 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:

$$V_\sigma(x_0, y_0) = \begin{cases} 0 & \text{if } \lambda_2 > 0 \\ \exp\left(-\frac{A^2}{2\beta^2}\right) \left(1 - \exp\left(-\frac{S^2}{2\gamma^2}\right)\right) & \text{otherwise} \end{cases} \quad (2.55)$$

where

$$A := |\lambda_1/\lambda_2| \quad \text{and} \quad S := \sqrt{\lambda_1^2 + \lambda_2^2} \quad (2.56)$$

and β and γ are tuning parameters. Before we discuss appropriate values for β and γ , we first seek to highlight the significance of eq. (2.55), and in particular, the ratios defined in eq. (2.56). A and S are known as the anisotropy measure and structureness measure, respectively.

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

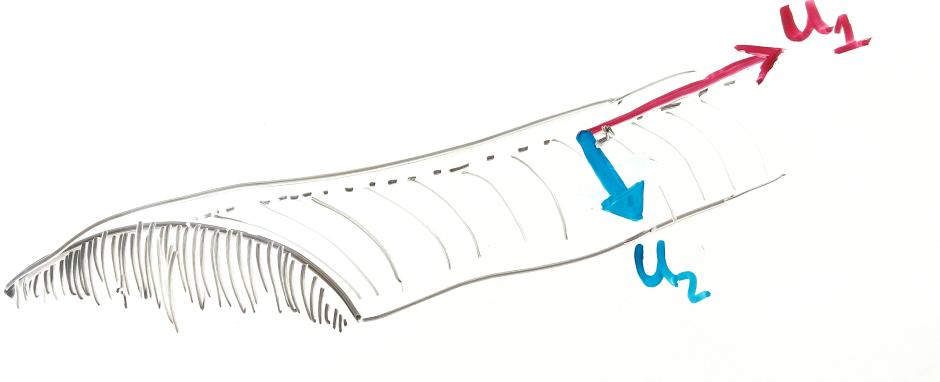


FIGURE 3: The principal eigenvectors at a ridge like structure

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 Note that the length of these vectors in this picture is not meant to represent their magnitudes, as u_2 should have a much larger relative magnitude by design!

Of course, if the the ridge is perfectly circular along its cross section (as was in section 2.2, 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 increasing steep. In such a case, λ_2 as a function of x would in fact be largest nearest to the bottom. This thought experiment should dispel a naive 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.

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, there will be points where we seem to "lose" our ridgelike structure due to issues with the sample section 4.1.

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

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 some of these issues by casting this as a multiscale problem in section 2.4.

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 7.

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 still 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 2-norm of the Hessian matrix. Thus the Frangi filter should also prefer areas of great curvilinear content in the image first of all.

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. (2.55) in which a factor of $\exp\{\dots\}$ and $(1 - \exp\{\})$ 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 rapidly decay in other situations.

Frangi further strengthened the filter by adding an additional case to in eq. (2.55), 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.

The Frangi vesselness filter: Choosing parameters β and γ

The parameters β and γ are meant to scale so that the peaks of $\exp\{\dots\}$ and $1 - \exp\{\dots\}$ coincide enough to be statistically significant but rapidly decay in areas not associated with curvilinear structure. What values of these parameters are appropriate is ultimately dependent on the context of the problem.

Frangi suggested for γ 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 approx the 2 norm of the Hessian. For β Frangi chose 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. If there is a lot of gain

then β should be smaller. If this is not the case, a stronger filter can be created by requiring A to be much smaller.

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

Linear Scale Space Theory

There is obviously a major disconnect in the ideas presented above. Although the ideas presented above require differentiation of continuous surfaces, our image is in fact a discrete pixel. 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. In particular, we seek to mitigate 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. Koenderink [6] 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 (and others) showed that a small set of intuitive axioms imply require that any such family of images must satisfy the heat equation

$$\Delta K(x, y, \sigma) = K_\sigma(x, y, \sigma) \text{ for } \sigma \geq 0 \text{ such that } K(x, y, 0) = u_0(x, y). \quad (2.57)$$

where $K : \mathbb{R}^3 \rightarrow \mathbb{R}$ and $u_0 : \mathbb{R}^2 \rightarrow \mathbb{R}$: is the original image (viewed as a continuous surface) and σ is a resolution parameter. Much work has been done to formalize this approach [11]. There is a long list of desired properties—we will try to identify a minimal subset of axioms and show that other desired properties follow.

Axioms

To make matters manageable, we require the one-parameter family of scaled images to be gener-

ated by an operation on the original image:

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

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

Linear-shift and Rotational invariance

Linear-shift (or translation) invariance means 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. Similarly, there should be not be favoritism toward any particular orientation of content within the image.

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 a nonzero real number (as opposed to an integer). Moreover, we require that the operator behaves continuously with respect to the scale parameter.

An argument from functional analysis (see [3]) implies that there is a so-called “infinitesimal generator” A which is a limit case of our desired operator T ; that is

$$Au = \lim_{\sigma \downarrow 0} \frac{T_\sigma u - u}{\sigma}$$

and moreover that there is a resultant differential equation concerning the derivative of the family and A :

$$\partial_\sigma K(x, y; \sigma) = \lim_{\sigma \downarrow 0} \frac{K(\cdot; \sigma + h) - K(\cdot; \sigma)}{h} = A(T_\sigma u) = A(K(\cdot, \sigma)) \quad (2.58)$$

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.

$$T_\sigma u = T_{\sigma_1 + \sigma_2} u \quad (2.59)$$

Causality condition

This requirement has great implication, and is also very successful in encoding our intuitive sense of “resolution”. 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.

In other words, if $K(x_0, y_0; \sigma_0)$ is a local maximum (at the point (x_0, y_0) , at this fixed σ_0) i.e. then an increase in scale can only weaken this peak, i.e.

$$\begin{cases} \nabla K(x_0, y_0; \sigma_0) = 0 \\ \Delta K(x_0, y_0; \sigma_0) < 0 \end{cases} \implies K(x_0, y_0; \sigma_1) \leq K(x_0, y_0; \sigma_0) \forall \sigma_1 \geq \sigma_0 \quad (2.60)$$

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.

$$\begin{cases} \nabla K(x_0, y_0; \sigma_0) = 0 \\ \Delta K(x_0, y_0; \sigma_0) > 0 \end{cases} \implies K(x_0, y_0; \sigma_1) \geq K(x_0, y_0; \sigma_0) \forall \sigma_1 \geq \sigma_0 \quad (2.61)$$

This implies that no image feature is sharpened by an decrease and resolution—the only result is a monotonic blurring of the image as scale parameter σ tends to infinity.s

Necessity of convolution

Where is this. **Uniqueness of the Gaussian Kernel**

Proved by Koenderink [1984] and Babaud [1986] that such an axiomatic foundation requires that

the family is generated by convolution with a kernel. The remainder of the development is to determine that the kernel is exactly the Gaussian.

To this, show that:

- a kernel satisfying the above axioms must satisfy the heat equation
- the gaussian kernel satisfies that.
- gaussian kernel is the only kernel that works (koenderink paper?)

$G_\sigma \star u_0$ solves the heat equation

given u_0 as a continuous image (unscaled), we construct PDE with this as a boundary condition.

$$u : \mathbb{R}^2 \supset \Omega \rightarrow \mathbb{R} \text{ with } u(\mathbf{x}, t) : \begin{cases} \frac{\partial u}{\partial t}(\mathbf{x}, t) = \Delta u(\mathbf{x}, t) & , t \geq 0 \\ u(\mathbf{x}, 0) = u_0(\mathbf{x}) \end{cases} \quad (2.62)$$

We show that

$$u(\mathbf{x}, t) = \left(G_{\sqrt{2t}} \star u_0 \right) (\mathbf{x}) \quad (2.63)$$

solves (the above tagged equation), where

$$G_\sigma := \frac{1}{2\pi\sigma^2} e^{(-|\mathbf{x}|^2/(2\sigma^2))}$$

First, we need a quick lemma regarding differentiation a continuous convolution.

Lemma 2.8. *Derivative of a convolution is the way that it is (obviously rewrite this).*

Proof. For a single variable,

$$\frac{\partial}{\partial \alpha} [f(\alpha) \star g(\alpha)] = \frac{\partial}{\partial \alpha} \left[\int f(t)g(\alpha - t)dt \right] \quad (2.64)$$

$$= \int f(t) \frac{\partial}{\partial \alpha} [g(\alpha - t)] dt \quad (2.65)$$

$$= \int f(t) \left(\frac{\partial g}{\partial \alpha} \right) g(\alpha - t) dt \quad (2.66)$$

$$= f(\alpha) \star g'(\alpha) \quad (2.67)$$

By symmetry of convolution we can also conclude

$$\frac{\partial}{\partial \alpha} [f(\alpha) * g(\alpha)] = f'(\alpha) * g(\alpha)$$

If f and g are twice differentiable, we can compound this result to show a similar statement holds for second derivatives, and then, given the additivity of convolution, we may conclude

$$\Delta(f * g) = \Delta(f) * g = f * \Delta(g) \quad (2.68)$$

□

Theorem 2.9. $u(\mathbf{x}, t) = \left(G_{\sqrt{2t}} * u_0 \right) (\mathbf{x})$ solves the heat equation.

Proof. We focus on the particular kernel

$$G_{\sqrt{2t}} = \frac{1}{4\pi t} e^{(-|x|^2/(4t))}$$

Then

$$\frac{\partial u}{\partial t}(\mathbf{x}, t) = \frac{\partial}{\partial t} \left(G_{\sqrt{2t}}(\mathbf{x}, t) * u_0(\mathbf{x}) \right) \quad (2.69)$$

$$= \frac{\partial}{\partial t} \left(G_{\sqrt{2t}}(\mathbf{x}, t) \right) * u_0(\mathbf{x}) \quad (2.70)$$

$$= \frac{\partial}{\partial t} \left(\frac{1}{4\pi t} e^{(-|x|^2/(4t))} \right) * u_0(\mathbf{x}) \quad (2.71)$$

$$= \left[-\frac{1}{4\pi t^2} e^{(-|x|^2/(4t))} + \frac{1}{4\pi t} \left(\frac{-|x|^2}{4t^2} \right) e^{-|x|^2/(4t)} \right] * u_0(\mathbf{x}) \quad (2.72)$$

$$= -\frac{1}{4t^2} \left(e^{(-|x|^2/(4t))} + |x|^2 G_{\sqrt{2t}}(\mathbf{x}, t) \right) * u_0(\mathbf{x}) \quad (2.73)$$

and from the previous lemma,

$$\Delta u(\mathbf{x}, t) = \Delta \left(G_{\sqrt{2t}} * u_0(\mathbf{x}) \right) = \Delta \left(G_{\sqrt{2t}} \right) * u_0(\mathbf{x})$$

We explicitly calculate the Laplacian of $G_\sigma(x, y) = A \exp(-\frac{x^2+y^2}{2\sigma^2})$ as follows:

$$\begin{aligned}
\frac{\partial}{\partial x} G_\sigma(x, y) &= A \left(\frac{-2x}{2\sigma^2} \right) \exp \left(-\frac{x^2 + y^2}{2\sigma^2} \right) \\
\implies \frac{\partial^2}{\partial x^2} G_\sigma(x, y) &= A \cdot \frac{\partial}{\partial x} \left[-\frac{x}{\sigma^2} \exp \left(-\frac{x^2 + y^2}{2\sigma^2} \right) \right] \\
&= A \left[-\frac{1}{\sigma^2} \exp \left(-\frac{x^2 + y^2}{2\sigma^2} \right) + \frac{x}{\sigma^2} \cdot \frac{2x}{2\sigma^2} \exp \left(-\frac{x^2 + y^2}{2\sigma^2} \right) \right] \\
&= A \exp \left(-\frac{x^2 + y^2}{2\sigma^2} \right) \left[-\frac{1}{\sigma^2} + \frac{x^2}{\sigma^4} \right] \\
&= \frac{1}{\sigma^2} G_\sigma(x, y) \left[\frac{x^2}{\sigma^2} - 1 \right]
\end{aligned}$$

By symmetry of argument we also may conclude

$$\frac{\partial^2}{\partial y^2} G_\sigma(x, y) = \frac{1}{\sigma^2} G_\sigma(x, y) \left[\frac{y^2}{\sigma^2} - 1 \right]$$

and so

$$\Delta G_\sigma(x, y) = \frac{\partial^2}{\partial x^2} (G_\sigma) + \frac{\partial^2}{\partial y^2} (G_\sigma) = \frac{1}{\sigma^2} G_\sigma(x, y) \left[\frac{x^2 + y^2}{\sigma^2} - 2 \right] \quad (2.74)$$

Then, given lemma 2.8, we conclude

$$\Delta [G_\sigma(x, y) \star u_0(x, y)] = \left(\frac{1}{\sigma^2} G_\sigma(x, y) \left[\frac{x^2 + y^2}{\sigma^2} - 2 \right] \right) \star u_0(x, y) \quad (2.75)$$

For particular choices of $\sigma(t) = \sqrt{2t}$ and $A = \frac{1}{4\pi t}$, we see

$$\Delta [G_{\sqrt{2t}}(x, y) \star u_0(x, y)] = \left(\frac{1}{2t} G_{\sqrt{2t}}(x, y) \left[\frac{x^2 + y^2}{2t} - 2 \right] \right) \star u_0(x, y) \quad (2.76)$$

$$= \left(G_{\sqrt{2t}}(x, y) \left[\frac{x^2 + y^2}{4t^2} - \frac{1}{t} \right] \right) \star u_0(x, y) \quad (2.77)$$

We then calculate the time derivative, using our particular choice of $\sigma(t) = \sqrt{2t}$ and $A = \frac{1}{4\pi t}$ as:

$$\frac{\partial}{\partial t} [G_{\sigma(t)}(x, y) * u_0(x, y)] = \frac{\partial}{\partial t} [G_{\sigma(t)}(x, y)] * u_0(x, y) \quad (2.78)$$

$$= \frac{\partial}{\partial t} [G_{\sqrt{2t}}(x, y)] * u_0(x, y) \quad (2.79)$$

$$= \frac{\partial}{\partial t} \left[\frac{1}{4\pi t} \exp \left(-\frac{x^2 + y^2}{4t} \right) \right] * u_0(x, y) \quad (2.80)$$

$$= \left[-\frac{1}{4\pi t^2} \exp \left(-\frac{x^2 + y^2}{4t} \right) + \frac{1}{4\pi t} \left(\frac{x^2 + y^2}{4t^2} \exp \left(-\frac{x^2 + y^2}{4t} \right) \right) \right] * u_0(x, y) \quad (2.81)$$

$$= \left(G_{\sqrt{2t}}(x, y) \left[\frac{x^2 + y^2}{4t^2} - \frac{1}{t} \right] \right) * u_0(x, y) \quad (2.82)$$

Combining these results, we find that

$$\frac{\partial}{\partial t} [G_{\sqrt{2t}} * u_0] = \Delta [G_{\sqrt{2t}} * u_0] \quad (2.83)$$

as desired. \square

We now return to our discussion of the Frangi filter. **A multiscale approach**

As identified earlier, scale is crucial in correctly identifying ridgelike structures. We wish to find regions that would receive a high vesselness score at any range, and consider them all together. Frangi [1] approached this problem by simply aggregating vesselness measure over all scales:

$$V(x_0, y_0) = \max_{\sigma \in \Sigma} V_\sigma(x_0, y_0) \quad (2.84)$$

where $\Sigma := \{\sigma_0, \sigma_1, \dots, \sigma_N\}$ is a range of scale spaces representative enough of all scales where meaningful content is expected to be found.

Thresholding

After this procedure, we are left with a matrix with as many samples/pixels as the original image, all with a vesselness measure between 0 and 1 for each pixel in the image:

$$V_{\Sigma} := [V(x,y)]_{\substack{0 \leq x < M \\ 0 \leq y < N}} \quad (2.85)$$

Whereas Frangi [1] left it relatively open-ended, if we wish to be final about the whole matter, we can ultimately say whether or not a pixel does in fact corresponds to a curvilinear structure or not. There are multiple methods of doing so. A naive yet enticing approach is to simply threshold past some certain point. The resulting matrix can be given in terms of either eq. (2.84) or eq. (2.85)

$$V_{\Sigma,\alpha}(x,y) = \begin{cases} 1 & \text{if } V(x,y) \geq \alpha \\ 0 & \text{else} \end{cases}, \quad \alpha > 0 \text{ fixed.} \quad (2.86)$$

First and Second Derivatives of Discrete Images

The way that you'd normally/naively describe it. Non multiscale. As in scipy or whatever.

Equivalent Methods of Calculating 2D Hessian

Note the following 6 methods should all theoretically be equivalent:

- convolve discrete image with a gaussian kernel, then take derivatives (no FFT). This is the “classical approach”
- Take derivatives of gaussian kernel, then convolve with the image/signal (again, no FFT)(due to)
- FFT image and FFT gaussian kernel then convolve in freq. space, then IFFT, then take derivatives in xy-space (my implementation)
- take derivatives of gaussian kernel, then FFT it, FFT the image, then colvolve in frequency space, then IFFT (slower due to size of large kernels)
- FFT image, then multiply by theoretical gaussian kernel in freq. space, then IFFT, then take derivatives in xy-space.

- FFT image, then multiply by theoretical gaussian kernel in freq space, then take derivatives in freq space, then IFFT.

Implementation Details: Convolution Speedup via FFT

As described above, the actual computation of derivatives is achieved via convolution with a gaussian. In practice, this is very slow for large scales. We instead, we perform a fast Fourier transform, which offers a speedup of N^2 operations vs $\mathcal{O}(n \cdot \log_2 n)$ operations

Fourier Transforms

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:

$$f(t) = \sum_{-\infty}^{\infty} c_n e^{i \frac{2\pi n}{T} t}, \quad c_n = \frac{1}{T} \int_{-T/2}^{T/2} f(t) e^{-i \frac{2\pi n}{T} t} dt \quad (2.87)$$

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

$$F(\mu) := \mathcal{F}\{f(t)\} = \int_{-\infty}^{\infty} f(t) e^{i 2\pi \mu t} dt \quad (2.88)$$

It can be shown that an inverse transform will then recover our original signal:

$$f(t) = \mathcal{F}^{-1}\{F(\mu)\} = \int_{-\infty}^{\infty} F(\mu) e^{i 2\pi \mu t} dt \quad (2.89)$$

Together, eq. (2.88) and eq. (2.89) are referred to as the *Fourier transform pair* of the signal $f(t)$.

Fourier Transform of a Discrete 1D signal

We wish to develop the Fourier transform pair for a discrete signal. 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 function (also referred to in DIP-GW as an impulse function), an infinite series of discrete impulses with equal spacing ΔT :

$$s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} \delta[t - n\Delta T], \quad \delta[t] = \begin{cases} 1, & t = 0 \\ 0, & t \neq 0 \end{cases} \quad (2.90)$$

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

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

$$\tilde{f}(t) = f(t)s_{\Delta T}(t) \quad (2.91)$$

From this (and actually the convolution theorem) we can calculate $\tilde{F}(t)$.

1D Discrete Fourier Transform (DFT)

Given the discrete signal \tilde{f} , we construct the transform $\tilde{F}(\mu) = \mathcal{F}\{\tilde{f}(t)\}$. by simply transforming the definition eq. (2.91).

$$\tilde{F}(\mu) = \sum_{n=-\infty}^{\infty} f_n e^{-i2\pi\mu n \Delta T}, \quad f_n = \tilde{f}(n) = f(n\Delta T) \quad (2.92)$$

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

-
-
-

2D Discrete Fourier Transform Convolution Theorem

Theorem 2.10 (2D DFT Convolution Theorem). *Given two discrete functions are sequences with the same length , that is: $f(x,y)$ and $h(x,y)$ for integers $0 < x < M$ and $0 < y < N$, we can take*

the discrete fourier transform (DFT) of each:

$$F(u, v) := \mathcal{D}\{f(x, y)\} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-2\pi i \left(\frac{ux}{M} + \frac{vy}{N}\right)} \quad (2.93)$$

$$H(u, v) := \mathcal{D}\{h(x, y)\} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} h(x, y) e^{-2\pi i \left(\frac{ux}{M} + \frac{vy}{N}\right)} \quad (2.94)$$

and given the convolution of the two functions

$$(f \star h)(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) h(x - m, y - n) \quad (2.95)$$

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

$$(f \star h)(x, y) = \mathcal{D}^{-1}\{MN \cdot F(u, v)H(u, v)\} \quad (2.96)$$

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

Proof.

$$(f \star h)(x, y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n)h(x-m, y-n) \quad (2.97)$$

$$= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \left(\sum_{p=0}^{M-1} \sum_{q=0}^{N-1} F(p, q) e^{2\pi i \left(\frac{mp}{M} + \frac{nq}{N} \right)} \right) \left(\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} H(u, v) e^{2\pi i \left(\frac{u(x-m)}{M} + \frac{v(y-n)}{N} \right)} \right) \\ (2.98)$$

$$= \left(\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} H(u, v) e^{2\pi i \left(\frac{ux}{M} + \frac{vy}{N} \right)} \right) \left(\sum_{p=0}^{M-1} \sum_{q=0}^{N-1} F(p, q) \left(\sum_{m=0}^{M-1} e^{2\pi i \left(\frac{m(p-u)}{M} \right)} \right) \left(\sum_{n=0}^{N-1} e^{2\pi i \left(\frac{n(q-v)}{N} \right)} \right) \right) \\ (2.99)$$

$$= \left(\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} H(u, v) e^{2\pi i \left(\frac{ux}{M} + \frac{vy}{N} \right)} \right) \left(\sum_{p=0}^{M-1} \sum_{q=0}^{N-1} F(p, q) \left(M \cdot \hat{\delta}_M(p-u) \right) \left(N \cdot \hat{\delta}_M(q-v) \right) \right) \\ (2.100)$$

$$= \left(\sum_{u=0}^{M-1} \sum_{v=0}^{N-1} H(u, v) e^{2\pi i \left(\frac{ux}{M} + \frac{vy}{N} \right)} \right) \cdot MNF(u, v) \quad (2.101)$$

$$= MN \cdot \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) H(u, v) e^{2\pi i \left(\frac{ux}{M} + \frac{vy}{N} \right)} \quad (2.102)$$

$$= MN \cdot \mathcal{D}^{-1} \{ FH \} \quad (2.103)$$

where

$$\hat{\delta}_N(k) = \begin{cases} 1 & \text{when } k = 0 \pmod{N} \\ 0 & \text{else} \end{cases} \quad (2.104)$$

□

Above, we make use of the following lemma

Lemma 2.11. *Let j and k be integers and let N be a positive integer. Then*

$$\sum_{n=0}^{N-1} e^{2\pi i \left(\frac{n(j-k)}{N} \right)} = N \cdot \hat{\delta}_N(j-k) \quad (2.105)$$

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

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

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

$$z^N - 1 = (z - 1)(z^{n-1} + \dots + z + 1) = (z - 1) \sum_{n=0}^{N-1} z^n. \quad (2.106)$$

thus giving us

$$0 = \left(e^{2\pi i (j-k)/N} - 1 \right) \sum_{n=0}^{N-1} e^{2\pi i n(j-k)/N} \quad (2.107)$$

To prove the claim in eq. (2.105), we consider two cases: First, if $j - k$ is a multiple of N , we of course have $e^{2\pi i n(j-k)/N} = (e^{2\pi i})^{n(j-k)/N} = 1$ and thus the left side of eq. (2.105) reduces to

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

In the case that $j - k$ is *not* a multiple of N , we refer to eq. (2.107). 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. (2.104), giving us the result.

□

FFT

As noted, the above result applies to the Discrete Fourier Transform. As noted, we actually achieve a convolution speedup using a Fast Fourier Transform (FFT) instead. We follow the developments of DIP-GW . 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:

$$F(\mu, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-i2\pi(\mu x/M + vy/N)} \quad (2.108)$$

$$= \sum_{x=0}^{M-1} e^{-i2\pi\mu x/M} \left[\sum_{y=0}^{N-1} f(x, y) e^{-i2\pi vy/N} \right] \quad (2.109)$$

$$= \sum_{x=0}^{M-1} e^{-i2\pi\mu x/M} \mathcal{F}_x\{f(x, y)\} \quad (2.110)$$

$$= \mathcal{F}_y\{\mathcal{F}_x\{f(x, y)\}\} \quad (2.111)$$

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 $x \in 0..M - 1$. 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 $(MN)^2$. (Note: 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

$$F(u) = \sum_{x=0}^{M-1} f(x) W_M^{ux}, \quad \text{where } W_m := e^{-i2\pi/m} \quad (2.112)$$

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. (2.112) and split it into odd and even entries in the summation

$$F(u) = \sum_{x=0}^{2K-1} f(x) W_{2K}^{ux} \quad (2.113)$$

$$= \sum_{x=0}^{K-1} f(2x) W_{2K}^{u(2x)} + \sum_{x=0}^{K-1} f(2x+1) W_{2K}^{u(2x+1)} \quad (2.114)$$

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

$$W_{(2m)}^{(2n)} = e^{\frac{-i2\pi(2m)}{2m}} = e^{\frac{-i2\pi m}{n}} = W_m^n \quad (2.115)$$

$$W_m^{(u+m)x} = e^{\frac{-i2\pi(u+m)x}{m}} = e^{\frac{-i2\pi unx}{m}} e^{\frac{-i2\pi mx}{m}} = e^{\frac{-i2\pi ux}{m}} (1) = W_m^{ux} \quad (2.116)$$

$$W_{2m}^{(u+m)} = e^{\frac{-i2\pi(u+m)}{2m}} = e^{\frac{-i2\pi ux}{2m}} e^{-i\pi} = W_{2m}^u e^{-i\pi} = -W_{2m}^u \quad (2.117)$$

Thus we can rewrite eq. (2.114) as

$$F(u) = \sum_{x=0}^{K-1} f(2x) W_{2K}^{2ux} + \sum_{x=0}^{K-1} f(2x+1) W_{2K}^{2ux} W_{2K}^u \quad (2.118)$$

$$\implies F(u) = \left(\sum_{x=0}^{K-1} f(2x) W_K^{ux} \right) + \left(\sum_{x=0}^{K-1} f(2x+1) W_K^{ux} \right) W_{2K}^u \quad (2.119)$$

The major advance comes via using the identities eq. (2.115) to consider the Fourier transform K frequencies later :

$$F(u+K) = \left(\sum_{x=0}^{K-1} f(2x) W_K^{(u+K)x} \right) + \left(\sum_{x=0}^{K-1} f(2x+1) W_K^{(u+K)x} \right) W_{2K}^{(u+K)} \quad (2.120)$$

$$\implies F(u+K) = \left(\sum_{x=0}^{K-1} f(2x) W_K^{ux} \right) - \left(\sum_{x=0}^{K-1} f(2x+1) W_K^{ux} \right) W_K^u \quad (2.121)$$

Comparing eq. (2.119) and eq. (2.121), 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 note them as follows:

$$\mathcal{D}_u\{f_{\text{even}}(t)\} := \sum_{x=0}^{K-1} f(2x)W_K^{ux} \quad (2.122)$$

$$\mathcal{D}_u\{f_{\text{odd}}(t)\} := \sum_{x=0}^{K-1} f(2x+1)W_K^{ux} \quad (2.123)$$

If we're calculating an M point transform (i.e. we're wishing to calculate $F(1), \dots, F(M)$), once we've calculated the first K discrete frequencies (i.e. $F(1), \dots, F(K)$) we may simply reuse the two values we've calculated in eq. (2.122) to calculate the next $F(K+1), \dots, F(K+K) = F(M)$. 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 $F(1), \dots, F(M)$. 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. (2.122) 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

The one caveat is that the above development was for transforming sequences whose lengths are perfect powers of 2. Since our inputs have no reason to be this, we need to adjust for this. The explanation is that you just do the part that's a power of 2 and then do the rest manually or pick a different power. See

The inverse DFT is actually a DFT of a modified function, see .

Alternative formulations of Scale Space Theory

Mention discrete treatments, mention relaxation of some of the dubious axioms.

Morphology

Methods of merging multiscale methods.

CHAPTER 3

IMPLEMENTATIONS

Calculating the Hessian

There are 6 ways to do this in theory.

To compare, standard gaussian calculation is implemented by `scipy.ndimage.filters.gaussian_filter`.

Scale Range detection

Find smallest and largest thing ...

Morphology: Plate Eroding

...

Morphology/Skeletonization

...

CHAPTER 4

RESEARCH PROTOCOL

Samples / Image Domain

We ultimately perform a PCSVN extraction on a set of 174 color placental images from a private database called NCS (from NYMH?). These are project files in GIMP which contain multiple layers. The layers together give a hand tracing of the vascular network and perimeter. A sample of overlaid layers in a representative sample is given in section 4.1

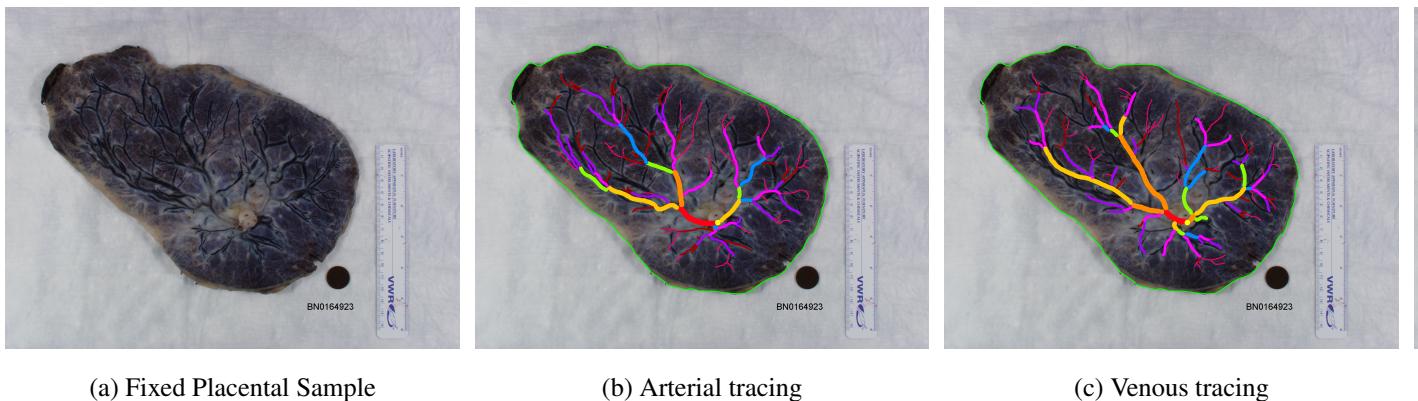


FIGURE 4: A representative placental NCS sample with vascular tracing

In fig. 4a, a cleaned, fixed placenta is shown. A detailed description of this procedure is given in fig. 4b and fig. 4c are both hand traces of the PCSVN, with a layer for each the arteries and veins. In our particular use case, there is no need to consider them separately, so we simply consider them together, as in fig. 4d. The coloration is meant to indicate the diameter of each vessel. There is also a cord insertion point notated, as well as the perimeter of the placental plate. These are hand-traced and rather labor intensive. A closer look at many of the samples often reveals some subjectivity in the tracings (often it's hard to see where the vein is, vascular networks are obscured, etc.)

For our procedure, we simply operate on the placental sample itself, without any understanding of its provided tracing except for comparing the strength of our algorithm. Of course, our goal is to



(a) Background Mask

(b) Sample with BG removed

(c) Grayscale

FIGURE 5: Preprocessed files from an NCS sample

develop an algorithm that can produce a “ground truth” tracing such as in fig. 4d or fig. 5d without any intervention.

For our purposes however, we will use the provided placental perimeter (shown in green in section 4.1. In developing a fully automated algorithm, it would be relatively straightforward to obtain this boundary ourselves using an Active Contour Model or perhaps even any edge finding algorithm followed by morphological / watershedding as in .

To build a sample suitable for use in our algorithm from section 4.1 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.

These procedures are performed automatically on the 174 image 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. (There are actually 201 images but 27 of them have mislabeled layers and were not autoprocessed correctly)

Image Preprocessing

As a point of technicality, the grayscale image in fig. 5c is not actually produced directly by the extractor plug-in, but created when the 3 channel RGB image fig. 5b is imported at the start of the algorithm. This grayscale conversion is simply done for ease of analysis on the sample: although the Frangi filter is designed for arbitrary dimension input [?], 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 [5], or “luminance” transform:

$$L = \frac{299}{1000} R + \frac{587}{1000} G + \frac{114}{1000} B \quad (4.1)$$

[‘] 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 of some interest to create an algorithm that, say, calculates a gradient or performs a convolution by a “reflection” across an arbitrary closed boundary (as opposed to the edge of the image matrix), we opted instead to simply exclude affected areas from consideration, and zero unwanted background pixels to speed up computation. This excluding function, `plate_morphology.dilate_plate`, ultimately relies on two functions provided by the Python library `scikit-image` [12]. The first, `skimage.segmentation.find_boundaries()`, takes the mask input (such as fig. 5a) and calculates where differences in a morphological erosion and dilation occur. That boundary itself is then dilated by the desired factor. The second is a “sparse” implementation of binary dilation that is particularly efficient for our problem. An array of indices of the image where the

`??` doesn’t really show what I want it to, but this is what it would look like. Repeat with a smaller border. Maybe the issue doesn’t occur with these as much? In the image above, $\sigma = 3$ and border radius is 80 and all it does is get rid of the stupid natural boundary, not a weird frangi response. Which is important in itself, but I was having an error just between the edge of the image.

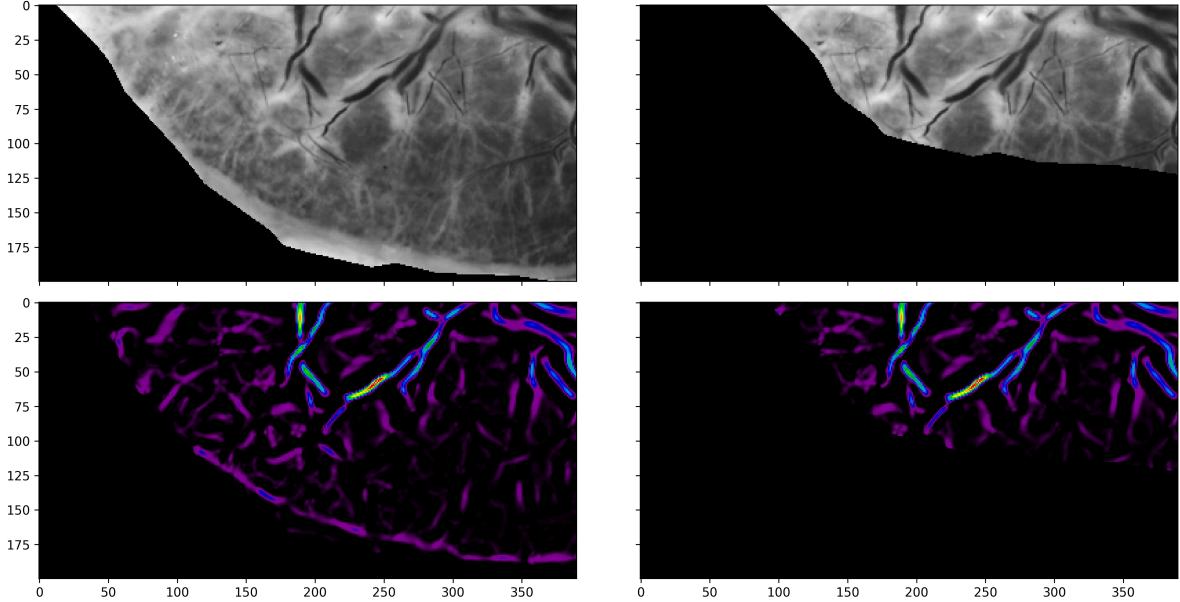


FIGURE 6: Demonstration of boundary dilation

The code for the above can be found by running `plate_morphology.py` as a top-level script (that is, within the “`if __name__ == __main__`” block of the file).

Multiscale Setup

Our multiscale Frangi filter requires a list of scales at which to probe. Each scale is chosen to accentuate features of a particular size, i.e. vessels of a particular radius. This list of scales is denoted as $\Sigma := \{\sigma_1, \sigma_2, \dots, \sigma_N\}$.

The smallest one should be an effective size where details are expected to be found, and the largest should be an effective size as well. In fact, following , it is reasonable to select these logarithmically; that is, for some selected inputs $m < M$ we have

$$\sigma_1 = 2^m, \sigma_j = 2^{(m + \frac{M-m}{N-1}j)}, \sigma_N = 2^M \quad (4.2)$$

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 filter will respond better at its particular scale,

but there are diminishing returns as σ increases. While the filter's response may vary substantially between, say $\sigma = 2$ and $\sigma = 3$, there will be not be a substantial difference in response between, say, $\sigma = 46$ and $\sigma = 47$. There was an earlier benefit as well, that is still worth mentioning for historical reasons. Previously, computing the vesselness measure was very expensive, and thus it was simply not feasible to collect so many large scale readings. This is moot with the development of FFT-based Frangi filter.

If there is no particular care taken in selecting a minimum and maximum range at which to probe, then we should assure that there is no noise being introduced at either ends, especially if the Frangi filter at which “throw out” bad ones somehow. We will approach this issue in our discussion of “variable thresholding” see section .

Convolve this via fft transform to get L_{σ_i}

Applying Vesselness Measure

Calculate the Hessian matrix of and then the eigenvalues using the function `hfft.fft_hessian`.

Scale-space post-processing

Multiscale Merging

Cleanup/Postprocessing

Measurements

NOTE DUMP

Erode plate / dilate boundary

Our function considers the placenta as a nonzero surface but the surface outside is zero (or, in many situations, masked). We're currently not implementing any way to “reflect” along the border, so instead the second degree behavior of the surface there will be incorrect in an area proportional to the scale size.

Describe how that function works. Earlier efforts are wrong, whatever.

The area which is affected should be larger than just the standard dropoff the gaussian however,

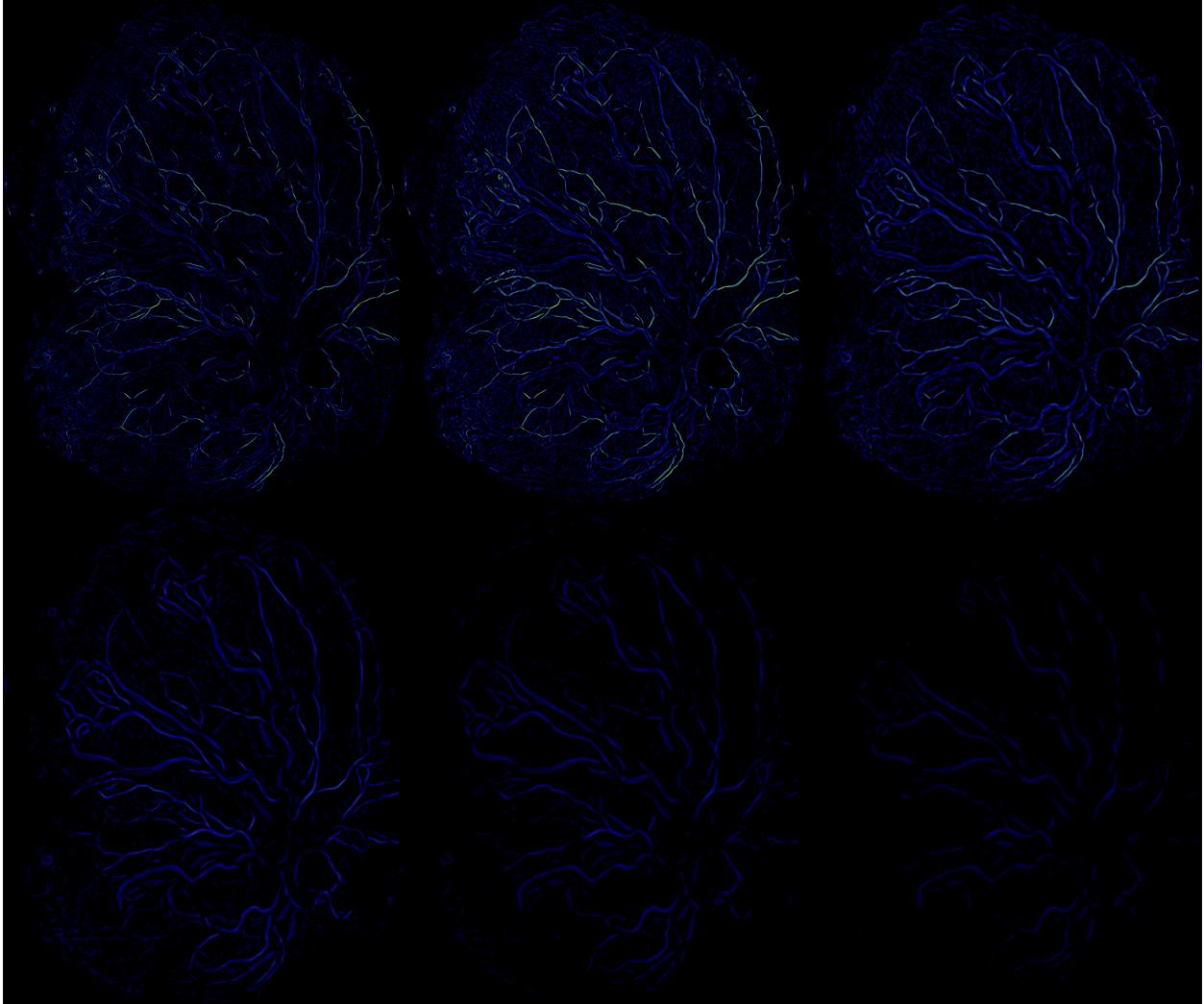


FIGURE 7: Frangi vesselness score at several scales

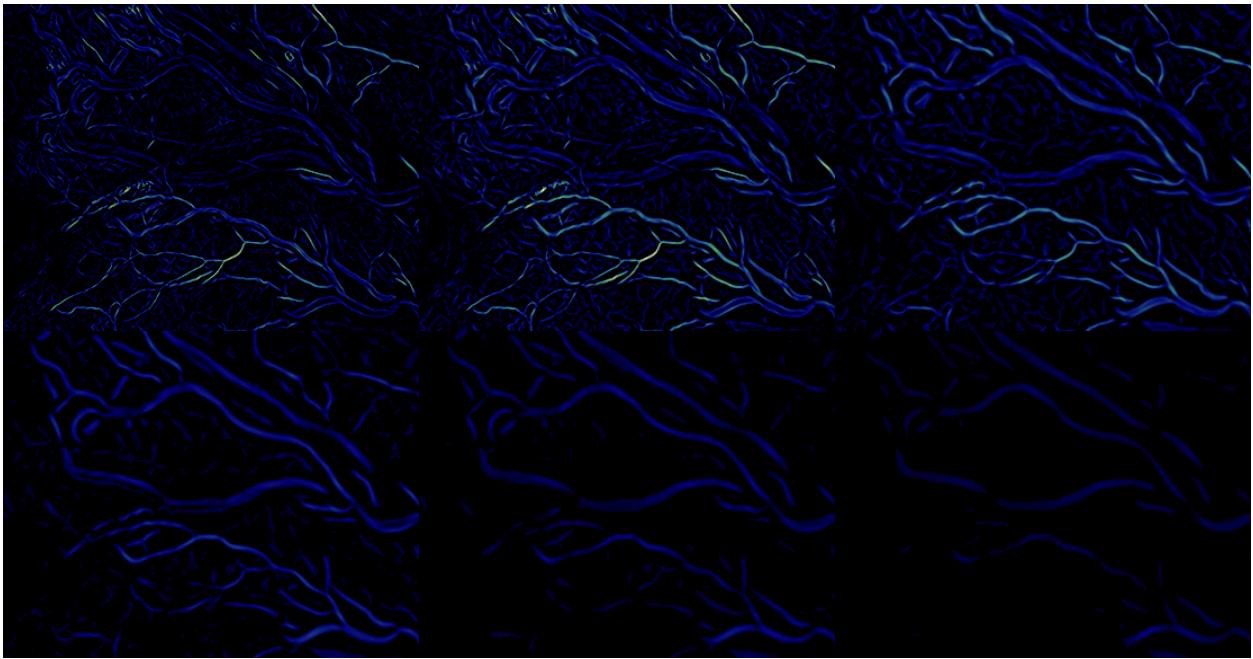


FIGURE 8: Frangi vesselness score at several scales (inset)

since we're interested in second derivative information.

Code Listings

CHAPTER 5

RESULTS AND ANALYSIS

Sample visual output

The confusion matrix

A Source of “False Negatives” in the NCS data set

Sometimes the output doesn't agree with the trace.

1. Collar is stupid.
2. Vessels suck sometimes.
3. Umbilical cord insertion point is stupid
4. Small vessels aren't accounted for.

Results

Answer Research Questions

CHAPTER 6

CONCLUSION

What it did well. What it didn't. People who want to expand—where should they start?

CHAPTER 7

FUTURE RESEARCH DIRECTIONS

Combine with ridge search. Use this as a preprocessing for future work. Better use of scales.

APPENDICES

APPENDIX A
APPENDIX TITLE

The following python files were written for are also available on github.com/wukm/pycake.git
and were developed with the following packages:

**matplotlib, version 2.02 scikit-image, version 0.13.0 numpy, version 1.12.0 scipy,
version 0.19.0**

BIBLIOGRAPHY

BIBLIOGRAPHY

- [1] Alejandro F Frangi, Wiro J Niessen, Koen L Vincken, and Max A Viergever. Multiscale vessel enhancement filtering. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 130–137. Springer, 1998.

Starting point.

- [2] Rafael C Gonzalez and Richard E Woods. Digital image processing prentice hall. *Upper Saddle River, NJ*, 2002.

Check this citation, find a better one.

- [3] E. Hille and R.S. Phillips. *Functional Analysis and Semi-groups*. American Mathematical Society: Colloquium publications. American Mathematical Society, 1957.

cited within Sporring just for one thing

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

- [5] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. [Online; accessed †today‡].

- [6] Jan J. Koenderink. The structure of images. *Biological Cybernetics*, 50(5):363–370, Aug 1984.

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

Could use the 3rd edition as well.

[8] 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 Jocelyne Troccaz, Eric Grimson, and Ralph Mösges, editors, *CVRMed-MRCAS'97*, pages 233–242, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.

[9] Sílvia Delgado Olabarriaga, M Breeuwer, and WJ Niessen. Evaluation of hessian-based filters to enhance the axis of coronary arteries in ct images. In *International Congress Series*, volume 1256, pages 1191–1196. Elsevier, 2003.

Comparison of Hessian-based vesselness enhancement filters (which they call HBVF). Introduction is very good. Also note, they use “exponentially distributed” scales for each filter! I don’t necessarily agree with their results though, and I don’t see what images they’re using. They do some parameter searching though, if I can understand their figures it might be useful to do myself in the thesis.

[10] Yoshinobu Sato, Shin Nakajima, Nobuyuki Shiraga, Hideki Atsumi, Shigeyuki Yoshida, Thomas Koller, Guido Gerig, and Ron Kikinis. Three-dimensional multi-scale line filter for segmentation and visualization of curvilinear structures in medical images. *Medical image analysis*, 2(2):143–168, 1998.

This was cited in Frangi I believe as a different Hessian filter. Look into this!
Gets compared in Olabarriaga, above. I am ignorantly assuming that this is strictly inferior to Frangi’s filter. Worth looking into.

[11] Jon Sporring. *Gaussian Scale-Space Theory*. Kluwer Academic Publishers, Norwell, MA, USA, 1997.

[12] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. scikit-image: image processing in Python. *PeerJ*, 2:e453, 6 2014.