Abstract—In this paper, we present an algorithm for enhancing neuronal structure from 3D Confocal Microscopy Images. Our algorithm computes a multi-scale phase congruency value at every pixel from a 3D image, which assigns values that indicate the presence of image features such as edges and lines. The phase congruency of a 3D image is calculated by carefully combining the convolutions of the image with a quadrature filter bank, so we leverage this information to supplement phase features. We analyse the outputs of the quadrature filter bank to enhance neuronal structure. We compare our method with the Hessian based enhancement of neuronal structure to demonstrate the advantages/efficacy of our algorithm.

Technical Areas: H8, Image and Video Segmentation; H11 Image and Video Enhancement/Filtering; H13, Wavelets; F4, Biomedical Image Processing

I. INTRODUCTION

This work aims to develop an enhancement algorithm using Phase Congruency (PC) to demonstrate the efficacy of Phase Congruency for biological/biomedical enhancement/segmentation. This work is important as proper enhancement (and possibly thereafter segmentation) in biological/biomedical applications is crucial for diagnoses and gaining an understanding of the function of imaged objects, such as neurons. For example, in [6], the authors develop an algorithm to compare neuronal morphology. Their algorithm’s efficacy heavily depends on the accuracy of the underlying enhancement algorithm.

Our dataset of 3D Confocal Microscopy neuron images contain features that can malign an agreeable enhancement of a neuron. These features include non-uniform neuronal intensity distributions, clutter and blob-like features.

II. BACKGROUND

We endeavour to perform neuronal enhancement using PC. PC, as will be demonstrated, is an intensity invariant feature detection algorithm [2]. We compare our method with the Hessian based enhancement of neuronal structure introduced by Frangi et al. [3]. In that method, the Hessian of a smoothed 3D image \( I \), \( (H(I * G(x,y,z))^\dagger) \), is evaluated at every pixel:

\[
\begin{pmatrix}
\frac{\partial^2}{\partial x^2}(I * G(x,y,z)) & \frac{\partial^2}{\partial x \partial y}(I * G(x,y,z)) & \frac{\partial^2}{\partial x \partial z}(I * G(x,y,z)) \\
\frac{\partial^2}{\partial x \partial y}(I * G(x,y,z)) & \frac{\partial^2}{\partial y^2}(I * G(x,y,z)) & \frac{\partial^2}{\partial y \partial z}(I * G(x,y,z)) \\
\frac{\partial^2}{\partial x \partial z}(I * G(x,y,z)) & \frac{\partial^2}{\partial y \partial z}(I * G(x,y,z)) & \frac{\partial^2}{\partial z^2}(I * G(x,y,z)) \\
\end{pmatrix}
\]

where * denotes linear convolution. The eigenvalues of this matrix, \( \{\lambda_1, \lambda_2, \lambda_3\} \), which describe local second order structure (curvature), can indicate the presence of neuronal structure. The image is smoothed by a Gaussian filter at several scales (various values of \( \sigma \)). This is done so that features of differing size can be identified.

One of the shortcomings of this method is that it is not sensitive to non-uniform neuronal intensity distributions. We believe our method can overcome these shortcomings as PC is an intensity invariant feature detector.

III. PHASE CONGRUENCY

PC is a multi-scale intensity invariant feature detector first introduced by Kovesi [2]. It is based on the premise that image features are located where the components of the Fourier decomposition are maximally “in phase”\(^2\). This can be illustrated by viewing a graphic (Figure 1) of the Fourier series of a square wave overlaid with its first few sinusoidal components. It can be seen that at the rising and falling edges, the phases of the Fourier components are maximally resonant. The feature type at every point at which PC is calculated can be determined by the phase angle at which the

\(G(x,y,z) = \exp[-\frac{x^2+y^2+z^2}{2\pi\sigma^2}]\).

This means that all the phase angles of the Fourier decomposition are roughly equal.

Fig. 1: Fourier components of a Periodic Square Wave
Fourier components are phase congruent. For example, at the rising and falling edges, the sinusoids’ phases are maximally congruent at $2\pi n, \ m \in Z$, whereas, at the midpoints of the crests and troughs, they are maximally congruent at $\frac{n\pi}{2}, \ m \in Z$. It is at these points where PC peaks. So, we observe that PC occurs at locations where the overall deviation of each Fourier component’s phase from the mean phase is minimized (in other words, the standard deviation of the phase angles is minimized). This minimization also takes into account each Fourier component’s magnitude, so the mean phase is really a weighted mean phase to account for small/large magnitudes. So we write:

$$PC(x) = \max_{\theta \in [0,2\pi]} \frac{\sum_n A_n(\cos(n\omega x + \phi_n - \theta))}{\sum_n A_n}$$

(2)

where $x$ is the coordinate on the $x$-axis, $\omega$ is the fundamental frequency, $A_n$ is the $n$th Fourier component’s magnitude, $\phi_n$ is the $n$th Fourier component’s phase and $\theta$ is the weighted mean phase. This formula is normalized by $\sum_n A_n$ so that $0 \leq PC(x) \leq 1$. As a consequence, points with high/low PCs are assigned values near 1/0. This normalization to the unit interval implies PC is intensity invariant.

However, for a general signal it is impractical to take its Fourier transform as it is a global frequency analysis tool. Specifying locations with high PC necessitates localized frequency analysis so that frequency/phase information can be determined at those locations. Wavelets enable localized frequency analysis [7]. In the spatial domain, wavelets may be specified by two parameters, dilation ($a > 0$) and translation ($b$). For a (mother) wavelet $w(x)$, $w(ax + b)$ is also a (baby) wavelet. In the frequency domain, a wavelet is the rescaling of the mother wavelet such that each rescaling selects specific bandwidths. We can see this by observing that $\mathcal{F}(w(ax + b)(\omega)) = \frac{1}{a} e^{iab} \mathcal{F}(w(x))(\frac{\omega}{a})$, where $\mathcal{F}$ is the Fourier Transform.

Now, to compute PC, the wavelets must be linear and be an odd/even pair (in quadrature)$^5$ in the spatial domain at each scale. Linearity, in general, preserves phase and the second quality ensures that an amplitude and phase at each scale and location are computable. So, the signal is filtered with a linear multi-scale wavelet quadrature filter bank. This bank consists of odd/even filter pairs $(M_n^e, M_n^o)$ at each scale to compute amplitudes/phases. Using this filter bank, the amplitude, $A_n(x)$, and phase, $\Phi_n(x)$, at every position $x$ and scale $n$ are:

$$A_n(x) = \sqrt{(I(x) * M_n^e(x))^2 + (I(x) * M_n^o(x))^2}$$

(3)

$$\Phi_n(x) = \arctan(I * M_n^e(x), I * M_n^o(x))$$

(4)

where $I(x)$ is the signal. Note that $\Phi_n(x)$ is not used to calculate PC, however $\Phi_n(x)$ can be used to describe feature type.

In this article, we use the Log-Gabor filter [4] as our quadrature filter because it has zero DC component and the filter bandwidth can be set arbitrarily large$^7$. The functional form of the Log-Gabor filter, which only has a closed form in the frequency domain, is:

$$G(\omega)_{1D} = \exp \left[ - \frac{\log^2 \frac{\omega}{\omega_0}}{2 \log^2 \frac{\omega}{\omega_0}} \right]$$

(5)

where $\omega$ is frequency, $\omega_0$ is the center frequency and $\frac{\omega}{\omega_0}$ sets the filter bandwidth. The functional form of the Log-Gabor filter in $N$ dimensions is

$$G_{ND}(\omega_1, \ldots, \omega_N) = G(\omega_1)_{1D} \prod_{i=1}^{N-1} e^{-\frac{(\omega_i - \omega_{0i})^2}{2 \sigma_i^2}}$$

(6)

where $0 < \omega_1, \omega_2, \ldots, \omega_{n-2} < \pi$ and $0 < \omega_{n-1} < 2\pi$ are filter angles/orientations, $\{\theta_0\}_{i=1}^{N-1}$ is the angular center frequency, $\omega_r$ is the radial frequency, and $\{\sigma_i\}_{i=1}^{N-1}$ determine the filter’s angular bandwidth. We now rewrite PC:

$$PC(x) = \frac{\max[E(x) - T, 0]}{\epsilon + \sum_n A_n(x)}$$

(7)

where $E(x) = \sqrt{(\sum_n I(x) * M_n^e(x))^2 + \sum_n I(x) * M_n^o(x))^2}$ and $T$ is a noise estimate of the image. The parameter $T$ is included because the quadrature filter is naturally insensitive to noise. However, if the signal to noise ratio is adequate, the amplitudes at important locations will overcome the noise. The author of [2] describes a procedure to estimate $T$ from the wavelet filters, however, the detail is omitted as this parameter can be freely chosen. The number $n$ is the scale index and $\epsilon$ is a small number to prevent division by zero. For $N$ dimensional images, PC is written as:

$$PC_{ND}(\vec{x}) = \frac{\sum_{\vec{a}} \max[W_{\vec{a}}(\vec{x}) (E_{\vec{a}}(\vec{x}) - T_{\vec{a}}), 0]}{\epsilon + \sum_{\vec{a}} \sum_n A_{n,\vec{a}}(\vec{x})}$$

(8)

where $\{\vec{a} = [a_1, a_2, \ldots, a_{N-1}]\}$ is the discretization of the filter angle set $\{\theta_0\}_{i=1}^{N-1}$. The formula for PC is summed over angles/orientations because the wavelet filters are now specified at angles/orientations in the frequency domain. However, there is a new term, $W_{\vec{a}}(\vec{x})$, that is included. It is a penalty term that discourages high PC values over small bandwidths. In [2], Kovese introduces

$$s_{\vec{a}}(\vec{x}) = \frac{1}{S} \left[ \frac{\sum_{n=1}^{N} A_{n,\vec{a}}(\vec{x})}{\epsilon + \max_{\vec{a}} A_{n,\vec{a}}(\vec{x})} \right]$$

(9)

where $S$ is the number of scales. This term, $0 < s_{\vec{a}}(\vec{x}) < 1$, is close to 1 when a point is phase congruent over a large bandwidth. Therefore, the term $W_{\vec{a}}(\vec{x})$, a sigmoid function, is

$$\frac{1}{1 + e^{b(|s_{\vec{a}}(\vec{x}) - 0.5|)}}$$

where $h$ and $g$ control the ‘cut-off’ and ‘gain’ (sharpness of cut-off) parameters of the sigmoid function. This term enhances the value of $s_{\vec{a}}(\vec{x})$.

$^5$The bandwidth of the Log-Gabor filter is actually infinite. The bandwidth specified above is the 3-dB bandwidth.
IV. OUR APPROACH

To enhance neuronal structure, we create a matrix based on the orientations in the wavelet filter bank. Our approach is to consider an image’s PC values over particular orientations and its variation with respect to orientation. This will give special information about the image features present. At \((x, y)\), we have \(PC_{\theta_i}(x, y)\) and we wish to find an axis where PC is minimized such that in an orthogonal direction PC is maximized. The orientations, \(\theta_i\), are specified in the spatial domain. For 2D images, we form a positive semi-definite matrix called the phase congruency orientation matrix that is defined at every pixel \((x, y)\) as:

\[
P_{2D}(x, y) = \begin{pmatrix}
PC_{\theta_1}, xx(x, y) & PC_{\theta_1}, xy(x, y) \\
PC_{\theta_1}, yx(x, y) & PC_{\theta_1}, yy(x, y)
\end{pmatrix}.
\] (10)

Each of these terms describe the PC variation about a particular axis in either the \(x\) or \(y\) direction. \(P\)'s eigenvectors are the axes of maximum/minimum variation and the eigenvalues describe their variations. The eigenvector associated with the smallest eigenvalue should point in the direction of the neuronal structure and the other should be orthogonal to it. We use \(P\)'s eigenvalues, \(0 \leq \lambda_1 \leq \lambda_2\), to enhance neuronal structure with the following equation:

\[
V = \exp(-\frac{R^2_B}{2\alpha^2})\left(1 - \exp(-\frac{S^2}{2c^2})\right)
\] (11)

where \(R_B = \lambda_1/\lambda_3\) is large for blob-like structure and \(S = \sqrt{\lambda_1^2 + \lambda_3^2}\) is small in cluttered regions. \(\beta\) and \(c\) are parameters that affect the strength of the terms \(R_B\) and \(S\).

In 3D, \(P\) instead has nine terms:

\[
P_{3D}(\vec{x}) = \begin{pmatrix}
PC_{(\theta_1), xx(\vec{x})} & PC_{(\theta_1), xy(\vec{x})} & PC_{(\theta_1), xz(\vec{x})} \\
PC_{(\theta_1), yx(\vec{x})} & PC_{(\theta_1), yy(\vec{x})} & PC_{(\theta_1), yz(\vec{x})} \\
PC_{(\theta_1), zx(\vec{x})} & PC_{(\theta_1), zx(\vec{x})} & PC_{(\theta_1), zz(\vec{x})}
\end{pmatrix}.
\] (12)

where \(\{\theta_1, \theta_2\}\) are the orientations in the 3D spatial domain and \(\vec{x} = (x, y, z)\) is a pixel location. We use its eigenvalues \(0 \leq \lambda_1 \leq \lambda_2 \leq \lambda_3\), to enhance neuronal structure in a similar fashion with

\[
V = \left(1 - \exp(-\frac{R^2_A}{2\alpha^2})\right)\exp(-\frac{R^2_B}{2\beta^2})\left(1 - \exp(-\frac{S''^2}{2c^2})\right)
\] (13)

where \(R_A = \lambda_3/\lambda_3\) distinguishes between plate-like and line-like structure and \(R_B'' = \lambda_1/\sqrt{\lambda_2^2 + \lambda_3^2}\) and \(S'' = \sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}\) are defined similarly as before and \(\alpha\) is another free parameter.

V. RESULTS

For the image shown below, we set the number of scales \(n\) to 10, the number of both orientations, \(\theta_1\) and \(\theta_2\) to 5, we set the noise response term \(T_\sigma\) to 0.09 for all orientations and scales, \(\epsilon = 0.0001\) and finally, we set \(\alpha = c = 20\) and \(\beta = 10\). We compared our method to a Hessian based enhancement filter proposed by Frangi et al. [3]. A sample image from our image database and its enhancements are shown in Figure 2. We believe our enhancement algorithm has significant improvements in comparison such as stronger neuronal enhancement and reductions of clutter and blob-like features. To visualize the images, we used an open source software known as 3D Visualization-Assisted Analysis (V3D) [8] which was exclusively made to process and visualize 3D Biomedical Data. A test image with both types of enhancement is shown below.

We also perform semi-manual segmentation with the aid of the V3D software. We do so by establishing a ground truth of the coordinates of the medial axis of neurons in 4 test images. This is done by first marking landmark locations along the neuron and tracing curves between appropriate landmarks (between landmarks where a path should exist). These landmarks are saved and loaded into the enhanced images and the curves between appropriate landmarks are also traced. The metric we evaluate to compare the curve tracing coordinates in the enhanced images to the original image is the root mean squared error (RMSE) given in pixels. Below is a table of the RMSE values for each image:

<table>
<thead>
<tr>
<th>Image #</th>
<th>PC RMSE Error (pixels)</th>
<th>Hessian RMSE Error (pixels)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image 1</td>
<td>1.099</td>
<td>4.271</td>
</tr>
<tr>
<td>Image 2</td>
<td>3.008</td>
<td>37.25</td>
</tr>
<tr>
<td>Image 3</td>
<td>15.47</td>
<td>42.54</td>
</tr>
<tr>
<td>Image 4</td>
<td>24.19</td>
<td>27.15</td>
</tr>
<tr>
<td>Image 5</td>
<td>0.7968</td>
<td>3.336</td>
</tr>
</tbody>
</table>

Our results indicate that an automatic segmentation has the ability to perform better using our enhancement algorithm in comparison as the difference between RMSE errors for each image is significant. An occurrence which is not revealed in the RMSE calculations is the possibility of the V3D software being unable to trace a curve between two landmarks in an enhanced image. This happened due to a "break" in the neuron path. To address this problem we place additional landmarks to create paths to compensate for the break and our RMSE calculations do take this into account. However, this particular occurrence is seemingly a constant feature of the images that were enhanced via Hessian analysis. An example is shown below (Figure 3) from one of the images in our data set along with its enhancements. In this example, we show the images overlaid with its tracings. Notice how the software is unable to perform a trace in a part of the neuron. We attribute this break to the fact that Hessian based enhancement solely relies upon an image’s intensity values whereas our approach is not very reliant on these values.

VI. CONCLUSION

PC provides an alternative filtering/enhancement process to those prevalent in the literature. We showed that our method more sensitively filters noise and preserves neuronal structure. We are convinced this is true since PC is an intensity invariant and that this property boosts the SNR. Also, our method does not have a prohibitive computational cost which makes our approach attractive.

REFERENCES

Fig. 2: Sample Image and its Enhancements

(a) Sample Image
(b) Enhancement via PC
(c) Enhancement via Hessian

Fig. 3: Path Breaking Example

(a) Original Image
(b) Enhancement via PC with tracing
(c) Enhancement via Hessian with tracing
The terms in the phase congruency matrix are:

\[ PC(\theta_i),_{xx}(\vec{x}) = \sum_{\{\theta_i\}} PC(\theta_i)(\vec{x})(\sin^2 \theta_1 + \cos^2 \theta_2) \]

\[ PC(\theta_i),_{yy}(\vec{x}) = \sum_{\{\theta_i\}} PC(\theta_i)(\vec{x})(\cos^2 \theta_1 + \cos^2 \theta_2) \]

\[ PC(\theta_i),_{zz}(\vec{x}) = \sum_{\{\theta_i\}} PC(\theta_i)(\vec{x})(\cos^2 \theta_1 + \sin^2 \theta_1) \]

\[ PC(\theta_i),_{xy}(\vec{x}) = -\sum_{\{\theta_i\}} PC(\theta_i)(\vec{x}) \cos \theta_1 \sin \theta_1 \]

\[ PC(\theta_i),_{xz}(\vec{x}) = -\sum_{\{\theta_i\}} PC(\theta_i)(\vec{x}) \cos \theta_1 \cos \theta_2 \]

\[ PC(\theta_i),_{yz}(\vec{x}) = -\sum_{\{\theta_i\}} PC(\theta_i)(\vec{x}) \sin \theta_1 \cos \theta_2 \]

where \( PC(\theta_i),_{xy}(\vec{x}) = PC(\theta_i),_{yx}(\vec{x}) \) and \( PC(\theta_i),_{xz}(\vec{x}) = PC(\theta_i),_{zx}(\vec{x}) \) and \( PC(\theta_i),_{yz}(\vec{x}) = PC(\theta_i),_{zy}(\vec{x}) \). As stated before, \( \{\theta_i\} = \{\theta_1, \theta_2\} \) are the orientations specified in the wavelet filter bank in equation (8) and \( \vec{x} = (x, y, z) \) is a pixel location. For the 2D case, simply drop the terms with \( \theta_2 \). The equations then become:

\[ PC_{\theta_1},_{xx}(x, y) = \sum_{\theta_1} PC_{\theta_1}(x, y) \sin^2 \theta_1 \]

\[ PC_{\theta_1},_{yy}(x, y) = \sum_{\theta_1} PC_{\theta_1}(x, y)(\cos^2 \theta_1) \]

\[ PC_{\theta_1},_{xy}(x, y) = -\sum_{\theta_1} PC_{\theta_1}(x, y) \cos \theta_1 \sin \theta_1 \]

where \( PC_{\theta_1},_{xy}(x, y) = PC_{\theta_1},_{yx}(x, y) \) and \( (x, y) \) is a pixel location.