Automated Vasculature Extraction from Placenta Images

Nizar Almoussa[a], Brittany Dutra[a], Bryce Lampe[b], Pascal Getreuer[a], Todd Wittman[a], Carolyn Salafia[c] and Luminita Vese[d]

[a]Department of Mathematics, University of California, Los Angeles, U.S.A
[b]Harvey Mudd College, Claremont, U.S.A.
[c]Placental Analytics, LLC, Larchmont, U.S.A.

ABSTRACT

Recent research in perinatal pathology argues that analyzing properties of the placenta may reveal important information on how certain diseases progress. One important property is the structure of the placental blood vessels, which supply a fetus with all of its oxygen and nutrition. An essential step in the analysis of the vascular network pattern is the extraction of the blood vessels, which has only been done manually through a costly and time-consuming process. There is no existing method to automatically detect placental blood vessels; in addition, the large variation in the shape, color, and texture of the placenta makes it difficult to apply standard edge-detection algorithms. We describe a method to automatically detect and extract blood vessels from a given image by using image processing techniques and neural networks. We evaluate several local features for every pixel, in addition to a novel modification to an existing road detector. Pixels belonging to blood vessel regions have recognizable responses; hence, we use an artificial neural network to identify the pattern of blood vessels. A set of images where blood vessels are manually highlighted is used to train the network. We then apply the neural network to recognize blood vessels in new images. The network is effective in capturing the most prominent vascular structures of the placenta.

Keywords: vasculature extraction, placenta images, segmentation, line detector, image filters, neural networks.

1. INTRODUCTION

Understanding placental vasculature can help doctors identify where prenatal development diverged from normality. This could potentially help lead to earlier diagnoses of significant life-long diseases.[1,2] However, the process of identifying placental blood vessels is currently costly and time-consuming; an expert must hand-trace the vessels in each placental image. By automating the detection of blood vessels, we hope to increase efficiency and reduce costs for emerging placental research.

1.1 Overview of Methods

To our knowledge, there is no single characteristic that can be used to accurately identify a placental vessel, as there is significant variance between placentas. For example, some placentas are covered in red blood while others appear brown or grey; some have thick, pronounced vessels while others have thinner ones.

Rather than formulating an explicit model for placental vasculature, we account for this variability by computing numerous "features" for the image and then feeding those features to a *neural network*. Such a neural network must then undergo training, where it is given a ground truth (a hand-traced vascular network) and then iteratively determine weights for these features in order to better detect blood vessels. Once trained, a

network can then be *simulated* on additional placenta images in order to identify vessels. For a more thorough introduction to neural networks, see.[3]

Unless otherwise specified, all feature computations and network simulations are performed on the green channel of the placenta image. We do this because our initial inspections revealed that blood vessels appeared most distinct and recognizable in the green channel. Preliminary results also suggested that this was desirable over computing features using other channels or RGB-to-gray conversions.

2. PRE-PROCESSING

Before being analyzed, all placental images are pre-processed to ultimately improve the performance of subsequent algorithms.

Figure 1. An example of placenta region extraction. Left: one of the placenta images. Right: extracted placenta region.

We first extract the placenta region by applying a threshold and morphological operations to the green channel of a placenta image (shown in Fig. 1 left). We then crop the image to obtain the result in Fig. 1 right, thereby making future calculations more efficient.

The majority of our placenta images feature large patches of glare, an additional challenge for the vasculature segmentation. To remove the glare, we apply an inpainting approach. This method of glare removal is a simplified approach of the method in[4] and produces comparable results. First, bright spots are identified as those pixels with an intensity above a pre-determined threshold, which we take to be 80% of the maximum intensity. Second, a top-hat filter is applied, and additional thresholding then accurately identifies appropriate glare regions. Third, the regions are dilated by several pixels in order to place the region boundaries on pixels unaffected by glare. Finally, solving Laplace's equation (1) fills in the glare regions Ω, which produces satisfactory results as seen in Figure 2.

$$u_{xx} + u_{yy} = 0; \ u(x,y) = I(x,y) \ \forall x,y \in \partial\Omega, \tag{1}$$

where $I(x,y)$ are the pixel values of the original image, outside the glare regions. We found that performing glare removal prior to the cropping procedure is preferable, as otherwise some glare regions could be unintentionally cropped.

3. FEATURES

We decided to borrow the neural-net approach of Malik,[5] as explained in the later sections. Numerous features would be computed for a placenta and then later fed to a neural network to detect vessels.

Some features that were computed on placenta images are described next. Other features including variance, curvature, eigenvalues of the 2nd moment matrix,[5] gradient magnitude, gradient orientation[5] and background subtraction are not described here, since these are relatively straightforward.

Neural networks trained using only these more straightforward features were found to be inappropriate for vessel detection since, in most cases, networks that use these features simply detect vessel edges but not the interior of the vessels themselves. However, these characteristics were still included alongside more intricate features in the event that they might reveal any implicit vessel information.
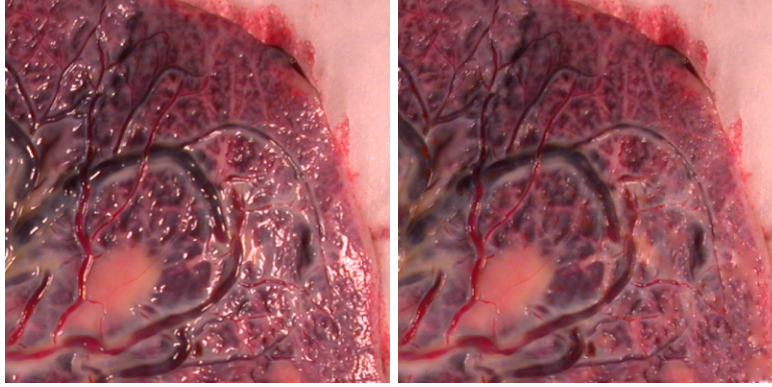
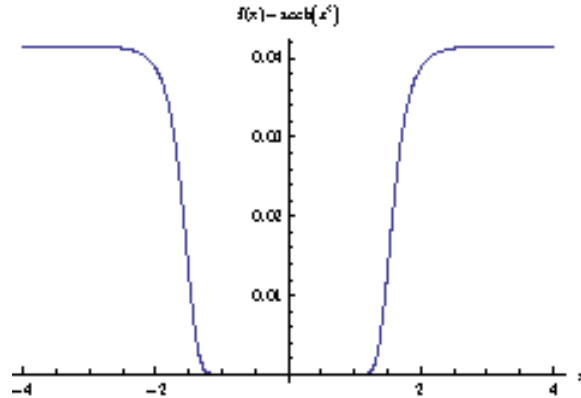Figure 2. A raw image is shown on the left, with glare removed on the right.

Figure 3. The error magnitude for $\text{sech}(x^5)$ and its 20th-order Padé approximant $f(x)$. Note from equation (2) that $|x| \leq 1/t$, since intensities can differ by at most ± 1.

3.1 Line Detectors

It was noted that in the green channel individual vessels show little variance in intensity after glare has been removed. Hence, when considering additional features to implement, we focused on those that could detect thick, uniform, curvilinear structures. We implemented several conventional line detectors, such as Steger's line detector,[6] a phase-coded detector,[7] a modified wide-line detector,[8] and the Hessian based vessel filter.[9] Some of these are used as additional features for our neural network. We give a brief overview of these detectors as well as any differences between our implementations and the literature.

3.1.1 Phase Coding

For this detector we use a complex-valued linear filter[7] with kernel

$$PC(x,\ y) = e^{2i \tan^{-1}(y-y_0,\ x-x_0)},$$

where x and y are coordinates within a circle of radius R inscribed within a square window; $(x_0,\ y_0)$ is the center of the window; values within this window but outside of the disk are taken to be 0. A potential drawback of this detector is that it will detect both bright and dark lines, and intersections are only weakly identified.

3.1.2 Wide-line Detector

This a more recent detector proposed by Liu *et al.*[8] that has seen success in similar medical contexts.[8,10] The detector consists of a nonlinear filter, where, for each pixel in the image, we tally the number of surrounding pixels within a radius R that differ in intensity by at most t when compared to the center pixel. However, to

smooth small intensity changes that occur near the threshold t, Liu *et al.*[8] actually tallies all pixels within this disk but weights them as

$$s(x, y, x_0, y_0, t) = \mathrm{sech}\left(\frac{(I(x,\ y) - I(x_0,\ y_0))^5}{t^5}\right), \tag{2}$$

where $I(x,\ y)$ is the intensity at $(x,\ y)$.

For computational efficiency, we use a 20th-order Padé approximant for $\mathrm{sech}(x^5)$ taken about $x_0 = 0$, namely

$$\mathrm{sech}(x^5) \approx \frac{15120 + (13x^{10} - 660)x^{10}}{15120 + (6900 + 313x^{10})x^{10}},$$

where x^{10} is computed from x using a logarithmic number of multiplications, as MATLAB's underlying C-based exponentiation function is highly inefficient. This approximant models the desired function very closely (as shown in Fig. 3) and exhibits significant speed improvement.

3.1.3 Steger's Detector

Steger's algorithm for thick line detection[6] is available online from the GRASP project[11] and is licensed under the GNU General Public License.[12] We incorporated this code into our MATLAB project by using Mex.

As provided, the Steger detector only gives a response at the center and on the edge of a thick line. Hence, to make this output more appropriate as a feature, these lines were filled in. At each pixel along the center of a line, pixels along the normals (for the appropriate width) were assigned a value of the line's "response," or the second-derivative of the line at that point, as described by Steger.[6] This yielded results surprisingly similar to the wide-line detector; the two methods largely agreed on the larger vessels, but differed more in the noisier, vessel-free regions of the placenta.

3.1.4 Modified Road Detection

We also proposed and implemented a novel modification to an existing road-detection technique[13] that made the method much more suitable for detecting vessels. Porikli uses a directional line filter to look for elongated rectangular regions in-between two homogeneous regions of different intensity levels on either side. This filter is somewhat limiting because it can only detect lines with a maximum thickness of five or six pixels, whereas blood vessels can be much thicker.

We multiply Porikli's directional line filter[13] by a Gaussian function in order to allow the filter to identify wider structures, with more weight towards the center. Our new filter is given as

$$g_\theta(i,\ j) = \cos\left(\frac{\pi i'}{2M}\right)\cos\left(\frac{\pi j'}{M}\right)e^{\frac{-(i^2+j^2)}{2(0.9M)^2}}, \tag{3}$$

where M is fixed, $i, j = -M,\ \ldots,\ M$, and i', j' are coordinates rotated by θ (we tested different values for the coefficient of M in the Gaussian function and found 0.9 to be the most effective for reducing noise while still retaining all the vessel information). We convolve this new filter over the entire image. Porikli's filter assumes that roads are generally brighter than their surrounding areas, but we found that most of the placenta images had blood vessels that were darker than the neighboring tissue, so we resolve this by inverting the convoluted image. We repeat this process for values of θ between 0 and π and combine the outputs for an overall result of

$$L = \left|\sum_{\theta \in \Theta} e^{2\pi i}(g_\theta \star I)^+\right| \tag{4}$$

where $\Theta = \{0,\ \pi/8,\ \ldots,\ 7\pi/8\}$ and I is the input image.

We found that using differently sized filters on the image could help detect small as well as large vessels, so we applied the filter with four different sizes of M, including 4, 7, 11, and 15. We created a simple noise reduction technique that labels all the connected segments and eliminates those with fewer pixels than a certain threshold, usually between 200 and 400 pixels. To combine the results from the small and large line filters, we
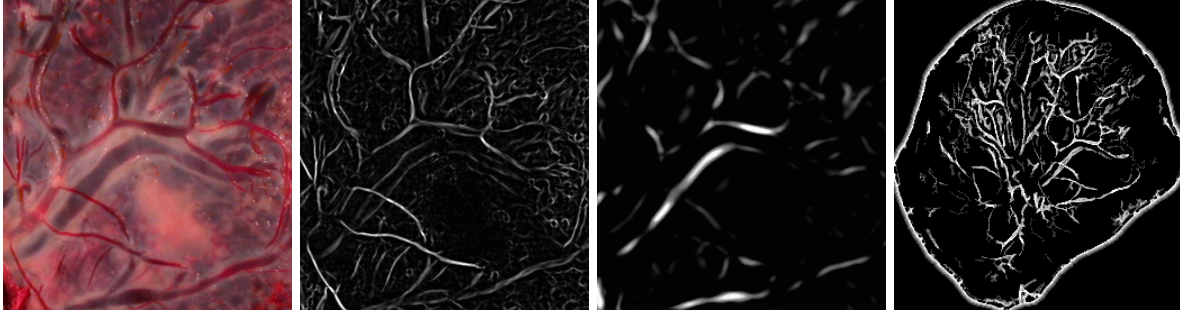
Figure 4. The original piece of the placenta (left) compared with the results from the smallest and largest road detection filters (middle). The rightmost image shows the modified road-detection feature on a whole placenta.

fuse together the four images. Again we can threshold the image to increase accuracy and reduce noise. Since there will only be four values in the normalized image, it is very simple to show points where a certain number of the four filters found a vessel. For example, by setting all points with a value of 0.25 to 0, the resulting image will show only points where two or more filters found a vessel, eliminating any accidental noise from one filter. Results of this filter are shown in Fig. 4.

Hessian-based Filter Frangi *et al.*[9] proposed a filter based on the eigenvalues of the Hessian matrix. Let $I(x, y)$ be the input image, and let

$$H_\sigma(x, y) = \left(G_\sigma * \left(\begin{array}{cc} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{array} \right) \right)(x, y),$$

where G_σ is a Gaussian filter. Vessels are identified as points where the eigenvalues λ_1, λ_2 of $H_\sigma(x, y)$ satisfy $\lambda_1 \approx 0$ and $\lambda_2 >> 0$.

We show in Fig. 5 results obtained using some of the above mentioned features, for comparison.
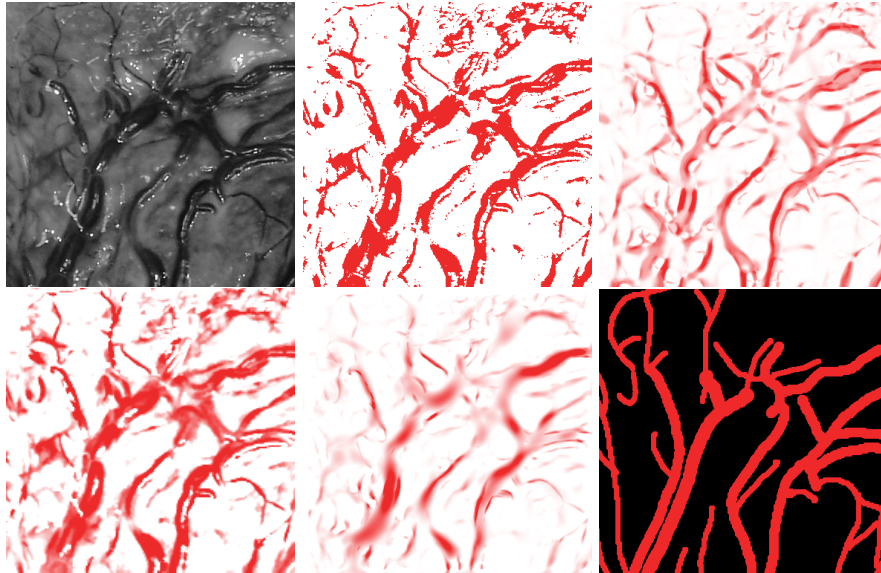

Figure 5. From left to right, top to bottom: original region of the placenta; vessels by background subtraction, line filter, wide line filter, Hessian-based filter, human.

We found that our enhanced road detection method was superior to the other line detectors when used on placental images. However, this method fails on blood vessels that are softer and brighter than their surrounding tissue. No single detector could sufficiently and reliably identify all blood vessels; and simple arithmetic and

morphological combinations of the detectors were also inadequate. Thus, we chose to borrow the neural-net approach taken by Malik;[5] the purpose of our neural network being to resolve the output from all of the features above, particularly the line detectors, while minimizing false-positives in the final result.

4. NEURAL NETWORK TRAINING

We use a set of manually traced images to train a neural network to detect blood vessels. Ideally, we would choose all pixels belonging to the placenta region to form the training set; however, our machines have limited memory, and the training time is preferred to be minimal. We overcome these problems by applying the following steps before training:

- Randomly pick n pixels from each traced image.

- For each pixel, compute its features and a binary value indicating whether it is a blood vessel.

- Randomly permute the sequence of pixels so that no single image can bias the training.

Manually tracing a vasculature network is subject to human interpretation. When we examined the traced images, we noticed that traced lines sometimes do not accurately cover vessels. This imprecision may affect the accuracy of the neural network output. To reduce the impact of these outliers, we added an option to transform the binary traced data to grayscale by convolving it with a Gaussian kernel, thereby giving greater weight to regions that had been traced while still allowing positive responses outside the traced areas. We also considered using mean-absolute error instead of mean-squared error when evaluating a network's performance, as it is more robust to outliers in the training data.

To determine the optimal combination of the features described in Section 3, numerous neural networks were trained with all possible combinations of 3-or-more features from all available features. This exhaustive search of the feature space was necessary because the processes of neural network training is not sufficient to determine which features are unneeded; determining those features to assign zero weight requires exponential time (since the greedy approach is not guaranteed to be optimal) but training occurs in polynomial time.

In addition, various parameters for the networks, such as the performance function (mean-squared error or mean-absolute error), the number of hidden nodes (5 or 15), how to normalize features (between $[0, 1]$ or $[-1, 1]$), and whether to apply a Gaussian blur to training data were also tested.

5. POST-PROCESSING

Our neural networks largely produced soft classifications of blood vessels, so further processing of these results was necessary. Grayscale neural-net outputs were thresholded to obtain a binary classification. These black-and-white images were then filtered for size; components smaller than usually 400 pixels were discarded as noise. Further filtering this by eccentricity (e.g., taking only highly eccentric, line-line objects), as suggested by,[10] was found to be unreliable. Due to the relatively high connectivity of our identified vessels, many components have low eccentricity, which is uncharacteristic of lines. Additionally, the edge is eroded since many of our detectors detected the edge of the placenta as a vessel.

In a few cases the neural network would produce an *almost* binary classification, with very few gray pixels. Thresholding was also applied in those cases.

6. MEASURING NETWORK PERFORMANCE

Once the neural-net output was post-processed, it remained to be shown which resulting binary classification was "best." Several metrics were used to compare the classifications to the traced data.

Between a given pixel P_1 from the target image and the corresponding pixel P_2 from the predicted image, four different conditions can occur:

- True Positive: both pixels P_1 and P_2 have a value of one.

- True Negative: both pixels P_1 and P_2 have a value of zero.

- False Positive: P_1 has a value of zero while P_2 has a value of one.

- False Negative: P_1 has a value of one while P_2 has a value of zero.

From these four conditions, we form the following binary classifications:

- TP: the total number of true positive pixels;

- TN: the total number of true negative pixels;

- FP: the total number of false positive pixels;

- FN: the total number of false negative pixels.

These binary classifications can be used to form different metrics for a network's performance. The ROC curve is one of these metrics. It is usually used to visualize the performance of different algorithms, but the ROC curves are not a good tool to choose the optimal algorithm.[14] To plot the ROC curves, two quantities are needed: the false positive rate

$$FPR = \frac{FP}{FP + TN}$$

and the true positive rate

$$TPR = \frac{TP}{TP + FN}.$$

The ROC curve is the set of all such pairs (FPR, TPR).

We can use the ROC curve to visually compare the performance of the neural-network technique versus individual features such as the wide-line and the road detection algorithms. Since the output of each algorithm is a grayscale image, a threshold is applied as described in Section 5 to obtain a binary image. This binary image and the image of manually traced blood vessels work as the predicted and target images, respectively. The threshold value is varied increasingly between 0 and 1. For each threshold value, FPR and TPR are computed from the binary classifications. The result is a ROC curve, as shown in Figure 7 left.

While the ROC curve is a good tool to visualize the performance of a network, it is not efficient in choosing the optimal algorithm, nor the best threshold to use for a particular network. A better metric is the Matthew Correlation Coefficient, or MCC, which puts more emphasis on the correlation between the two images rather than the actual accuracy in the prediction (Fig. 6). The MCC uses the binary classification as following

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

The area under the MCC curve was measured for each neural network, and the network with the largest area was chosen to be the optimal network. From the MCC curve (see Fig. 7 right), one can also compute the optimal threshold value. This value was used as the initial threshold value for the graphical user interface.

7. RESULTS AND DISCUSSIONS

A side-effect of the randomization of training data, described in Section 4, is that several networks trained using the same features can have variable performance. To account for this, we train several (usually 3) networks using the same features and the same training images. We then simulate *all* of these networks and average their results; this produces more connected vasculature than just taking a single network on its own. This is demonstrated in Figure 8.

Results of our methods are shown in Fig. 9. While not perfect, the neural network does identify many prominent vessels. This result is further improved when fused with the result from the Hessian-based filter. We note that the width of the detected vessels is more accurate than in the manual tracing.
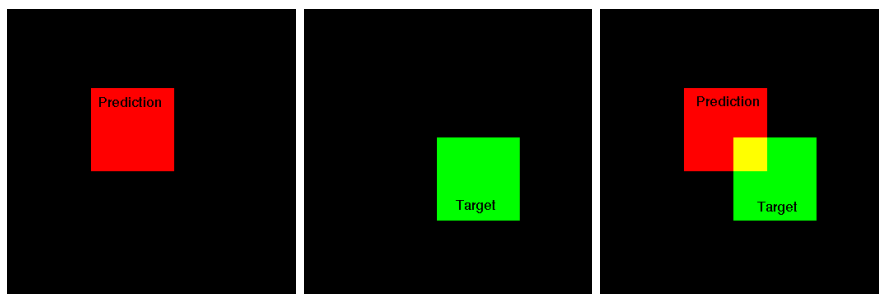
Figure 6. An example to motivate the MCC metric. A hypothetical neural-net prediction is shown in red, with a desired target in green. Although the prediction largely does not agree with the target, there is a strong correlation between the two shapes.
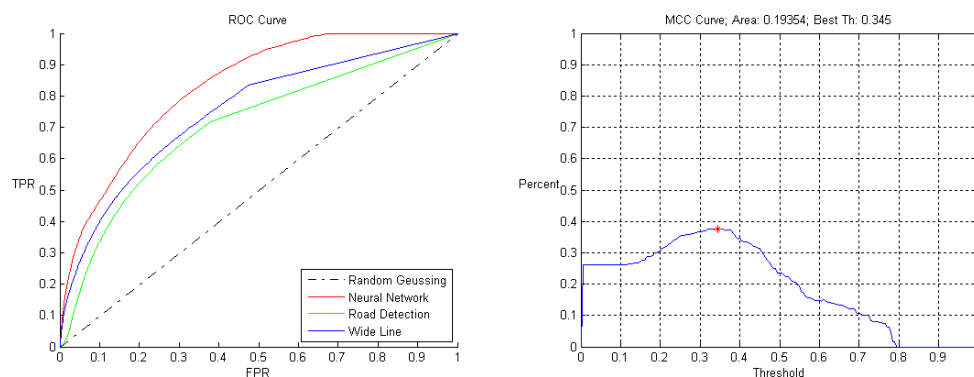

Figure 7. An ROC curve for a neural network is shown (left) and compared with individual features, and an MCC profile is also shown (right) for a different neural network. Neither of these networks are the same as the one used in Figure 9, but almost all networks exhibited similar performance.

Noise is present in the classifications, which is why we provide a feature in our GUI front-end for the user to erase classified vessels.

With 5 binary options and 8 features in all, there were $2^5 \sum_{k=3}^{8} \binom{8}{k} = 7008$ possible nets to generate. This number was reduced to approximately 2000 by eliminating the binary options that did not seem immediately beneficial after preliminary results (e.g., blurring training data and normalizing to $[-1, 1]$). However, due to hardware limitations we were only able to generate roughly 1292 of those 2000. Hence our results are somewhat incomplete, but still promising.
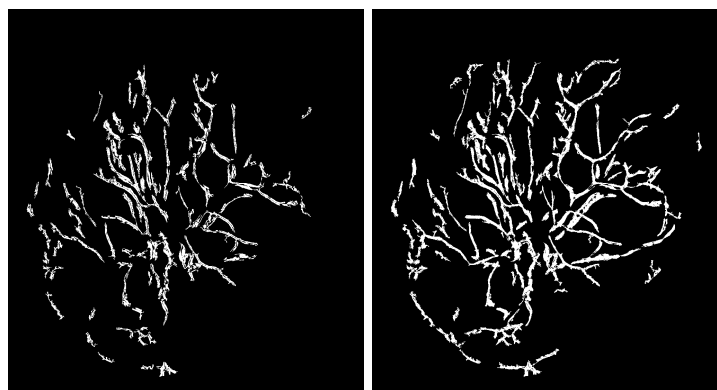

Figure 8. On the left is the result from one neural network. On the right is the average (and the post-processed) result from three neural nets trained using the same features and training data. The randomization of the training process seems to be mitigated by using several nets.
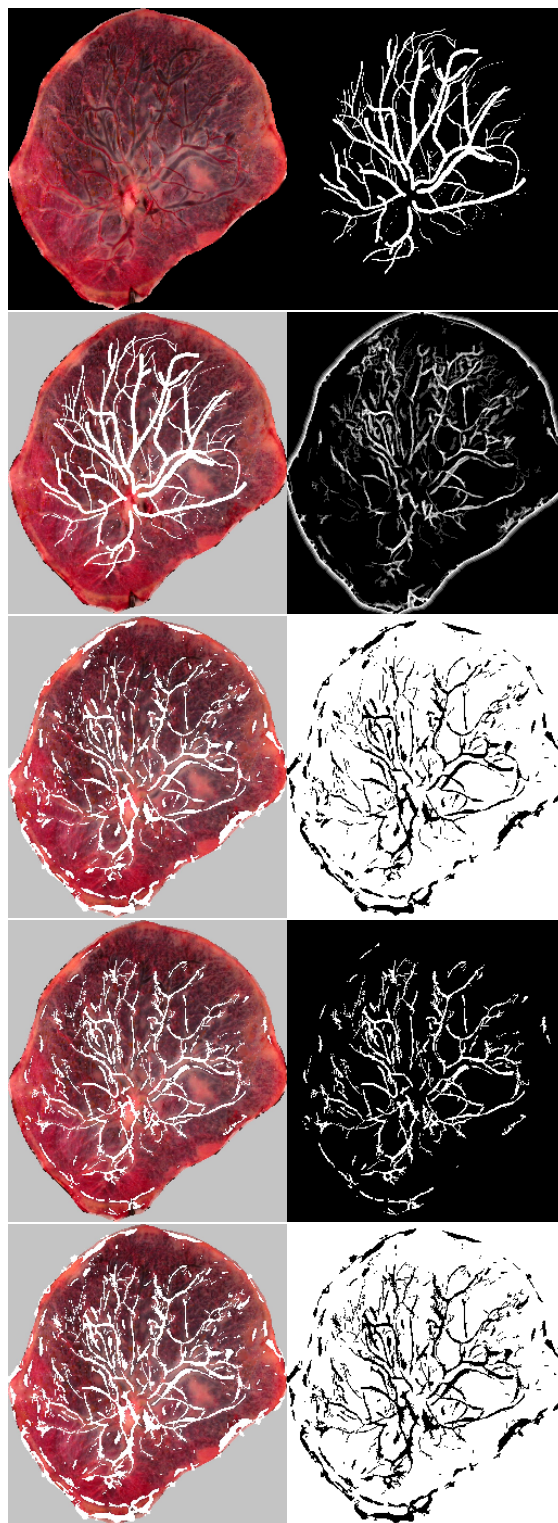
Figure 9. Left column, top to bottom: placenta image; manually-traced vessels and placenta; placenta and vessels by Hessian-based filter; placenta and vessels by neural networks; fused vessels from Hessian-based filter and neural network. Right column, top to bottom: manually-traced vessels; vessels by road detection algorithm; vessels by Hessian-based filter; vessels by neural network; fused vessels from Hessian-based filter and neural network.

With that incompleteness in mind, we found that the neural net with the highest area under its MCC curve (0.22) used: the gradient magnitude and gradient angle, the wide-line detector, the Steger detector, and our modified road detector. These results are further improved when fused with those obtained using the Hessian-based filter.

We found that, in-general, nets that used the mean-absolute error for their performance function performed slightly better than those that used mean-squared error. Blurring training data to reduce the impact of outliers had little effect on performance. Somewhat surprisingly, networks with only five hidden nodes performed better than networks with 15-or-more hidden nodes; they were also faster to train and simulate. We were unable to determine whether normalizing features to $[0, 1]$ or $[-1, -1]$ was preferable, although we use the range $[0, 1]$ in our GUI interface.

Input images were 1,200 by 1,600 pixels, and the computations to compute image features were completed within two minutes for each image using a 2.13GHz Core 2 Duo with 2GB of RAM without down sampling. Most of this time was spent computing the wide-line detector and our modified road detection feature. The time to train a neural network usually took 5 minutes, however only a subset (8 images) of all available images were used to train. Training with more images had little impact, possibly implying that the sampled points taken from those 8 images were abundant enough to be reflective of an "average" placenta's vascular network.

In conclusion, we think that our methods are effective at identifying blood vessels, and we are confident it can eventually be used to help facilitate placental research. With improved hardware, more networks and features can be used to obtain improved results.

ACKNOWLEDGMENTS

REFERENCES

[1] Yampolsky, M., Salafia, C., Shlakhter, O., Haas, D., B.Eucker, and Thorp, J., "Modeling the variability of shapes of a human placenta," *Placenta* **29**, 790–797 (2008).

[2] Salafia, C. and Yampolsky, M., "Metabolic scaling law for fetus and placenta," *Placenta* **30**, 468–471 (2008).

[3] Fausett, L., [*Fundamentals of Neural Networks*], Prentice-Hall (1994).

[4] Lange, H., "Automatic glare removal in reflectance imagery of the uterine cervix," *Proc. SPIE Medical Imaging* **5747**, 2183–2191 (2005).

[5] Martin, D., Fowlkes, C., and Malik, J., "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE PAMI* **26**, 530–549 (2004).

[6] Steger, C., "An unbiased detector of curvilinear structures," *IEEE TPAMI* **20**, 113–125 (1998).

[7] Clode, S., Zelniker, E., Kootsookos, P., and Clarkson, V., "A phase coded disk approach to thick curvilinear line detection," tech. rep., ARROW Discovery Service (2004).

[8] Liu, L., Zhang, D., and You, J., "Detecting wide lines using isotropic nonlinear filtering," *IEEE TIP* **16**, 1584–1595 (June 2007).

[9] Frangi, A., Niessen, W., Vincken, K., and Viergever, M., "Multiscale vessel enhancement filtering," *LNCS* **1496**, 130–137 (June 1998).

[10] Liu, L. and Zhang, D., "Extracting tongue cracks using the wide line detector," in [*ICMB*], 49–56 (2008).

[11] Allen, B., "Gravitational radiation analysis and simulation package." `http://www.lsc-group.phys.uwm.edu/~ballen/grasp-distribution/GRASP/include/` (January 1999).

[12] Inc., F. S. F., "GNU general public license." `http://www.gnu.org/copyleft/gpl.html` (June 2007).

[13] Porikli, F., "Road extraction by point-wise Gaussian models," in [*SPIE AeroSense Technologies and Systems for Defense and Security*], 758–764 (2003).

[14] Fawcett, T., "ROC graphs: Notes and practical considerations for researchers," 1–38 (March 2004).