Vessel Segmentation

Wenzhen Zhu
WUSTL

Notation

Color Coding

- 🟨 Equation
- 🟩 My thoughts and explanations
- 🟥 book segment with an error or an error description
- 🟧 answer for the problem
- 🟪 question
- 🟪 assumption
- 🟦 intermediate answer

Intro

ArchHacks 2016 `https://archhacks.io/` 's theme is HealthTech.
I participate this hackathons and after building an app and talking to several medical students, I realized that I can actually apply what I learned from computer vision class and make some really world application to help patients who are suffering and to help doctors to be more efficient. That's why I choose to do this project.

Edge detection is an essential task in computer vision, and vessel segmentation is also an important task in clinical care and medical research.

First, doctor can diagnose many disease from small change of vessels, like diabetes. Second, some disease needs to remove vessels first in order to make further diagnosis.

Helper function

Related Work

Morphology Operators

We define a 2D image whose range is $[N_{\min}, N_{\max}]$ as a functional $S: \mathbb{R}^2 \to [N_{\min}, N_{\max}]$ and a 2-D structuring element as a functional $B: \mathbb{R}^2 \to R$ where R is the set of neighborhood of origin.

We define basic operators, with respect to the structuring element B with scaling factor *e*, image *S* and point $M_0 \in \mathbb{R}^2$

Erosion

$$\epsilon_B^e\ (S)\,(M_0) = \text{Min}_{M \in M_0 + \epsilon B(M_0)}(S(M))$$

Dilation

$$\delta_B^e\ (S)\,(M_0) = \text{Max}_{M \in M_0 + \epsilon B(M_0)}(S(M))$$

Opening

$$\gamma_B^e\ (S)\,(M_0) = \delta_B^e(\epsilon_B^e(S))$$

Closing

$$\phi_B^e\ (S)\,(M_0) = \epsilon_B^e(\delta_B^e(S))$$

Top-hat

$$\text{TH}_B^e\ (S)\,(M_0) = S - \gamma_B^e(S)$$

Geodesic dilation

$$S^1 = \inf(\{\text{Max}_{M \in M_0 + c}(S_m(M))\},\ S(M_0))$$
$$S^{d+1} = \Delta_{S_m}^{d+1}\ (S)\,(M_0) = \Delta_{S_m}^1\left(S^d\right)(M_0)$$

Geodesic closing

$$\phi_{S_m}^{\text{rec}}\ (S) = N_{\max} - \gamma_{(N_{\max} - S_m)}^{\text{rec}}(N_{\max} - S)$$

Algorithms

```
dir = NotebookDirectory[];
img = ColorConvert[
   Import[dir <> "DRIVE/training/images/21_training.tif"], "Grayscale"];
imgData = ImageData[img, "Byte"];
```

Step 0: Construct L_i

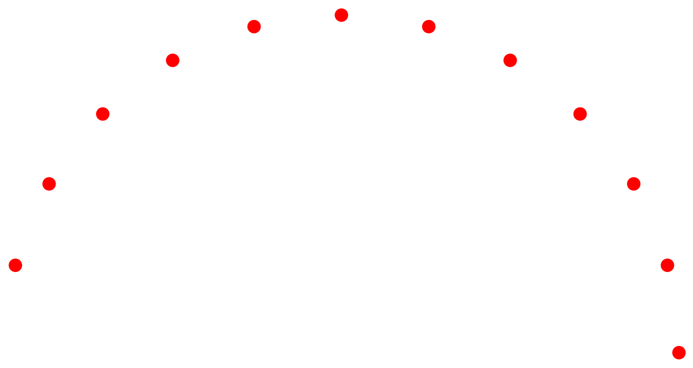
This paper is interesting because it used a linear structure element L_i which is 15-pixels long 1-pixel wide.

```
angles = Table[{Cos[t], Sin[t]}, {t, 0, 11 π / 12, π / 12}]
```

$\left\{\{1,0\},\left\{\frac{1+\sqrt{3}}{2\sqrt{2}},\frac{-1+\sqrt{3}}{2\sqrt{2}}\right\},\left\{\frac{\sqrt{3}}{2},\frac{1}{2}\right\},\left\{\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right\},\left\{\frac{1}{2},\frac{\sqrt{3}}{2}\right\},\left\{\frac{-1+\sqrt{3}}{2\sqrt{2}},\frac{1+\sqrt{3}}{2\sqrt{2}}\right\},\{0,1\},\left\{-\frac{-1+\sqrt{3}}{2\sqrt{2}},\frac{1+\sqrt{3}}{2\sqrt{2}}\right\},\left\{-\frac{1}{2},\frac{\sqrt{3}}{2}\right\},\left\{-\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right\},\left\{-\frac{\sqrt{3}}{2},\frac{1}{2}\right\},\left\{-\frac{1+\sqrt{3}}{2\sqrt{2}},\frac{-1+\sqrt{3}}{2\sqrt{2}}\right\}\right\}$

For every $\frac{\pi}{12}$, there is an linear structure element.

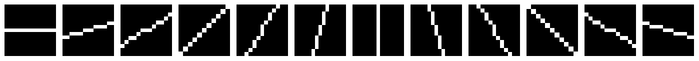
```
Graphics[{Red, PointSize → Large, Point[angles]}]
```



```
linearStrut =
  Table[Round[{Cos[t], Sin[t]} * #] &/@ (Range[15] - 8), {t, 0, 11 π / 12, π / 12}];
```

Here is a visualization of the structure element.

```
GraphicsRow[showBinaryImage[kernelFromStructure[#]] &/@ linearStrut]
```



```
kernels = kernelFromStructure[#] &/@ linearStrut
```

Step I

Calculating S_{op}

The first step is calculating S_{op}, which can be summarized in the followings:
1. Opening with 12 structure element on S_0(original images)
2. Pick the maximum from these 12 images pixel-wise and form a maximum image
3. Do geodesic opening on the maximum image, the result will be our S_{op}

$$S_{op} = \gamma_{S_0}^{rec}\ (\mathrm{Max}_{i=1,..,12}\{\gamma_{L_i}(S_0)\})$$

```
openingImageList = Opening[img, #] &/@ kernels
```

```
AbsoluteTiming[openingDataList = ImageData[#] & /@ openingImageList]
```

```
maxOpeningData = MapThread[Max, openingDataList, 2]
```

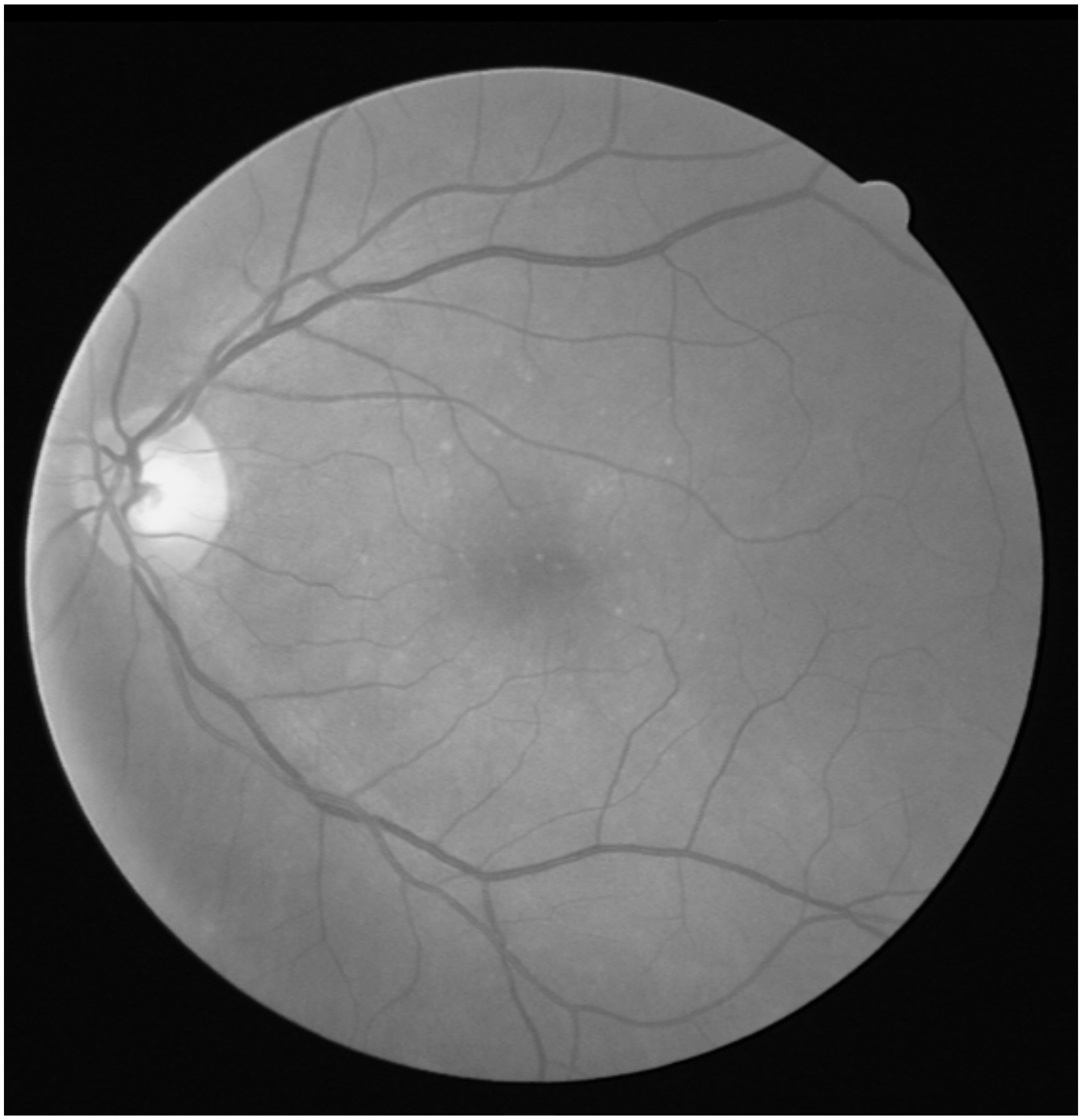
```
maxOpeningImage = Image[maxOpeningData]
```

```
markerImage = img;
mask = maxOpeningImage;
```

```
SopData = geodesicOpening[markerImage, mask]
```

```
SopImage = Image[SopData]
```

Here is a visualized result S_{op} image.



Step 2

Calculating S_{sum}

This transformation (a sum of top hats) reduces small bright noise and improves the contrast of all linear parts. Vessels could be manually segmented with a simple threshold on S_{sum}

$$S_{\text{sum}} := \sum_{i=1}^{12} (S_{\text{op}} - \gamma_{L_i}(S_0))$$

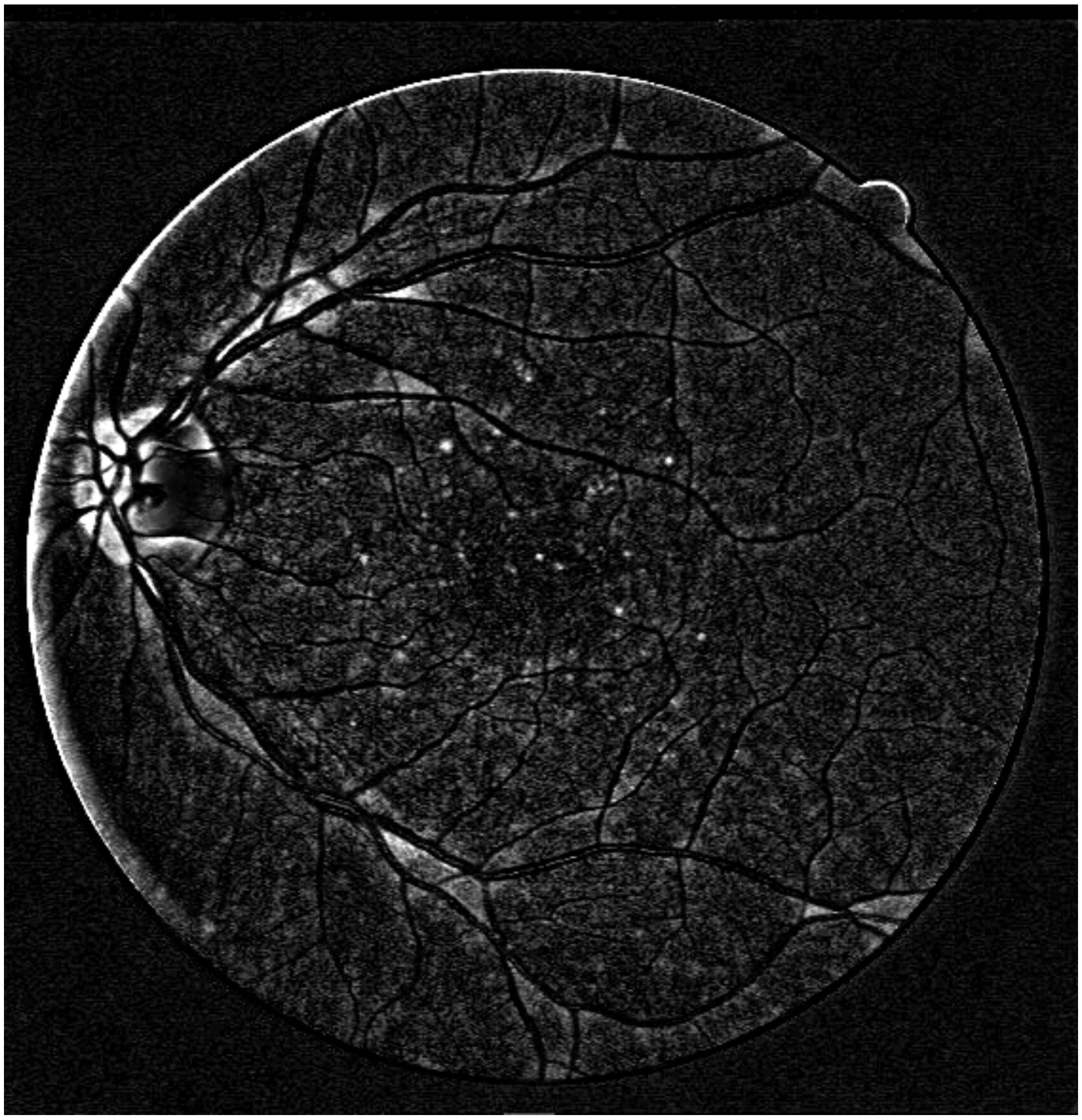
The second step is calculating S_{sum}, which can be summarized in the followings:

1. Opening with 12 structure element on S_0(original images).
2. Use S_{op} to minus the opening images pixel-wise.
3. Sum the 12 difference image up to get S_{sum}image

```
AbsoluteTiming[topHatData = (SopData - #) & /@ openingDataList]
```

```
AbsoluteTiming[SsumData = Total[topHatData]]
```

```
SsumImage = Image[SsumData]
```



Step 3

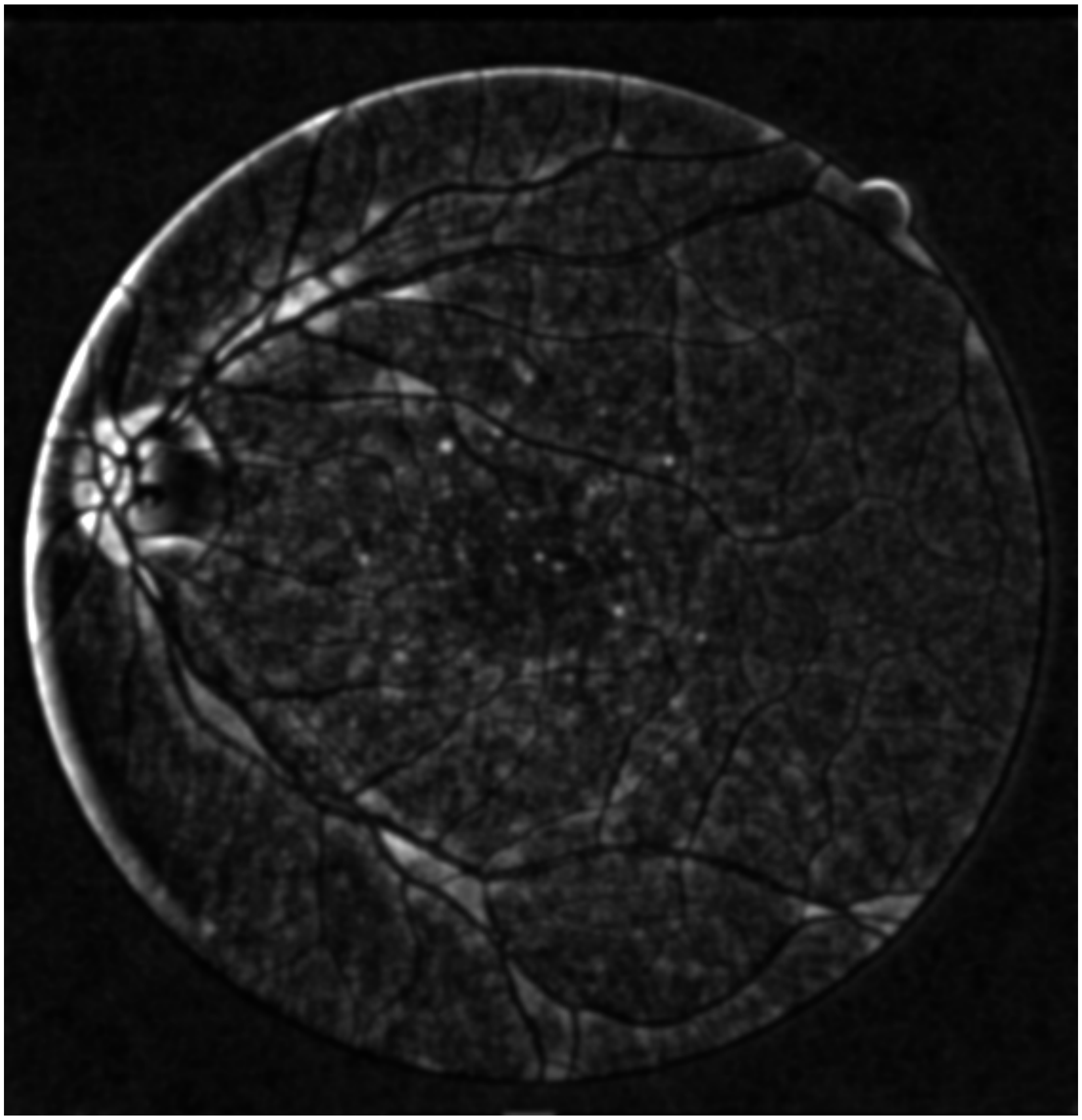
This step is very straight forward:

1. Apply Gaussian filter with $r = 7$ and $\sigma = 7/4$
2. Apply Laplacian filter with $r = 1$.

$$S_{\text{lap}} = \text{Laplacian}\left(\text{Gaussian}_{\sigma = 7/4}^{\text{width} = 7\text{ px}}(S_{\text{sum}})\right)$$

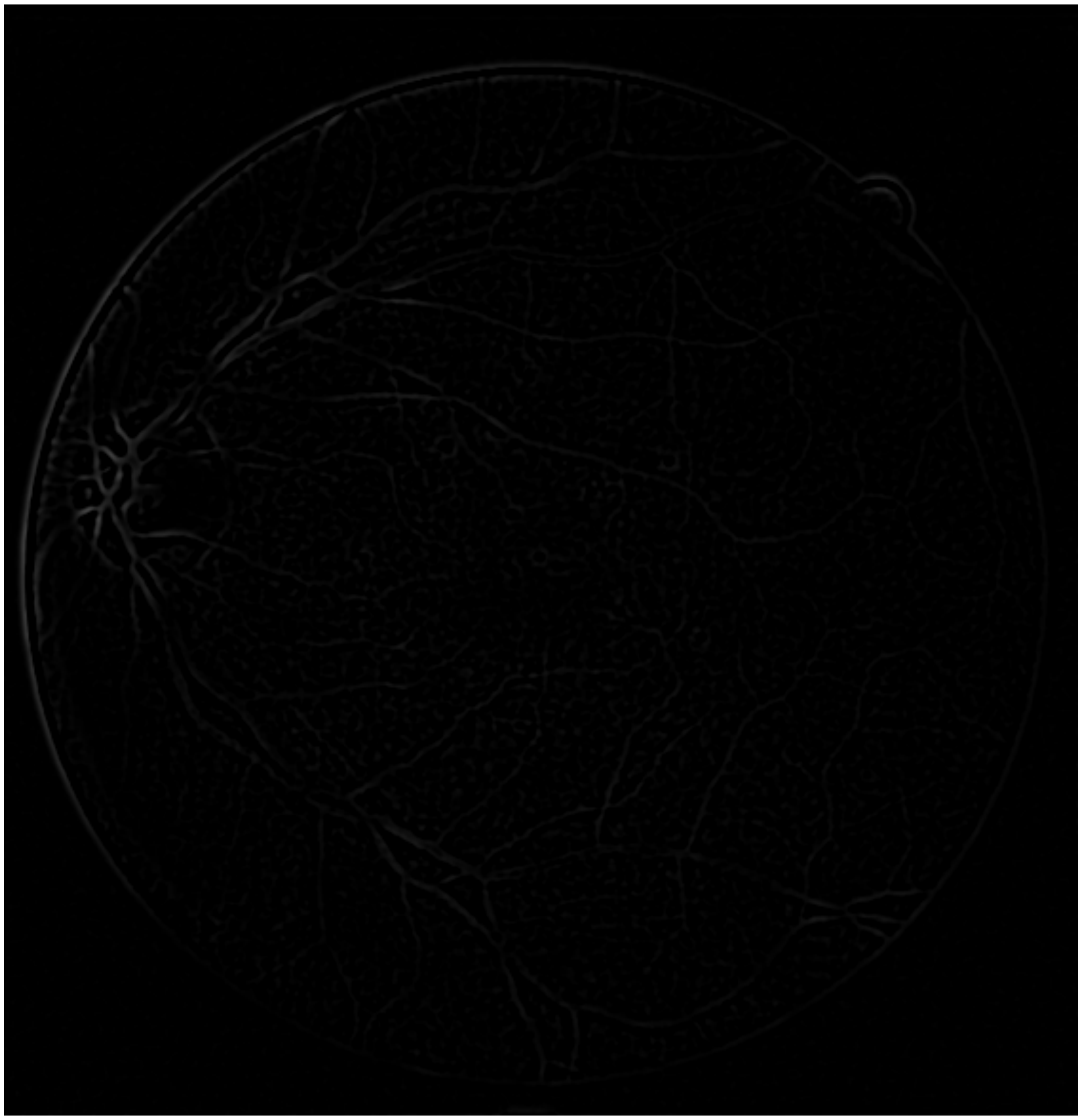
Applying Gaussian filter

```
gaussianImage = GaussianFilter[SsumImage, {7, 7 / 4}]
```



Applying Laplacian filter

```
SlapImage = LaplacianFilter[gaussianImage, 1]
```



Step 4

Calculating S_1

This step can be summarized in the followings:
1. Opening with 12 structure element on S_{lap}(Laplacian applied image).
2. Pick the maximum from these 12 images pixel-wise and form a maximum image
3. Do geodesic opening on the maximum image, the result will be our S_1

$$S_1 = \gamma^{\text{rec}}_{S_{\text{lap}}}\left(\text{Max}_{i=1,\ldots 12}\{\gamma_{L_i}(S_{\text{lap}})\}\right)$$

```
AbsoluteTiming[openingOnSlapData = ImageData /@ (Opening[SlapImage, #] & /@ kernels)]
```

Pick the maximum

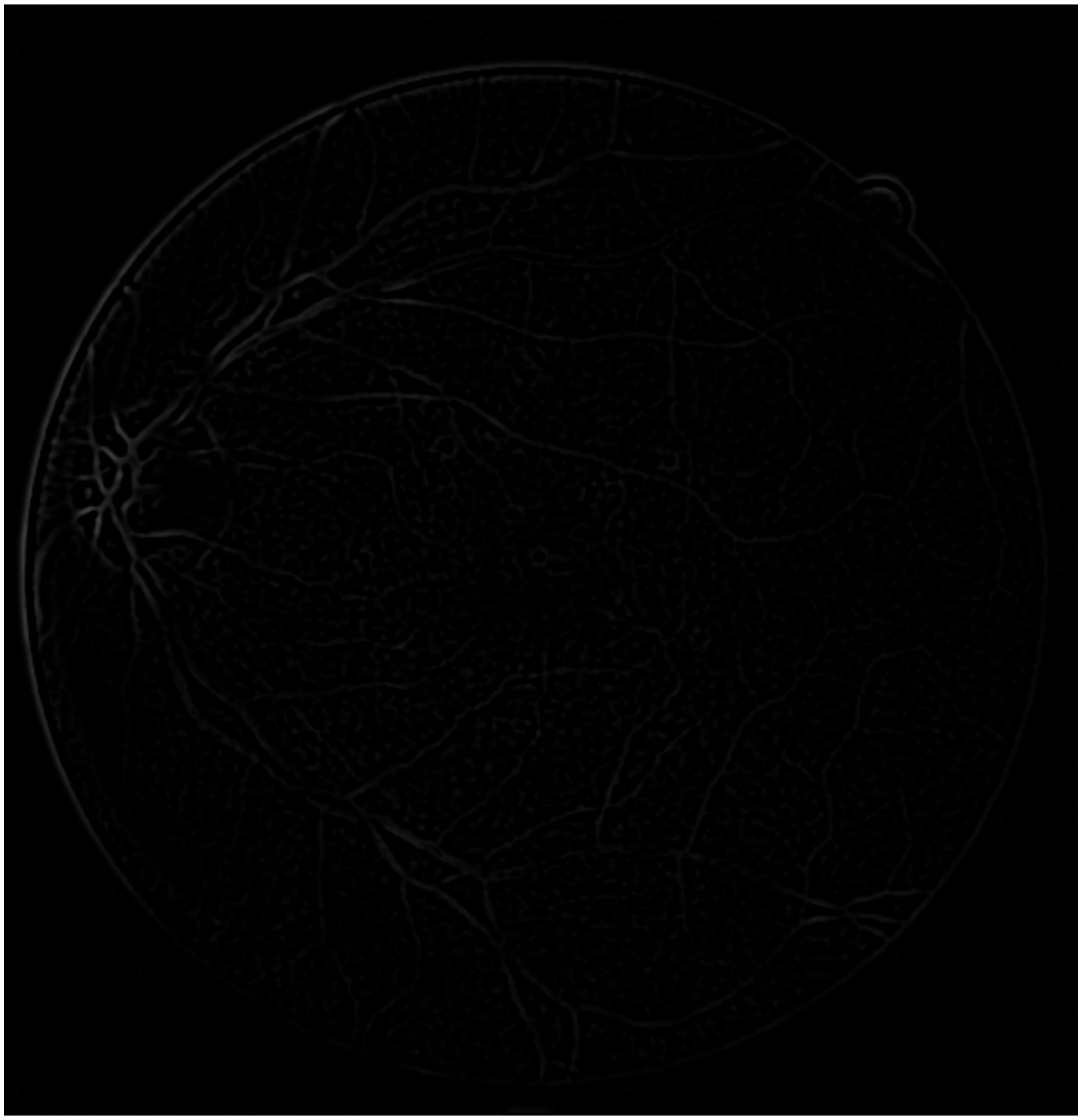
```
maxOpeningOnSlapData = MapThread[Max, openingOnSlapData, 2];
```

```
maxOpeningOnSlapImage = Image[maxOpeningOnSlapData]
```

Apply Geodesic opening on the maximum

```
S1Data = geodesicOpening[SlapImage, maxOpeningOnSlapImage]
```

```
S1 = Image[S1Data]
```



Calculating S_2

This step can be summarized in the followings:
1. Closing with 12 structure element on S_1.
2. Pick the minimum from these 12 images pixel-wise and form an image.

3. Do geodesic closing with marker being S_1, mask being the minimum image.

$$S_2 = \phi^{\text{rec}}_{S_1}\ (\text{Min}_{i=1,\dots,12}\{\phi_{L_i}(S_1)\})$$

```
AbsoluteTiming[closingImageList = Closing[S1, #] & /@ kernels]

AbsoluteTiming[closingDataList = ImageData /@ closingImageList]

minClosingData = MapThread[Min, closingDataList, 2];

minClosingImage = Image[minClosingData]

S2Data = geodesicClosing[S1, minClosingImage]

S2 = GeodesicClosing[S1, 1]
```

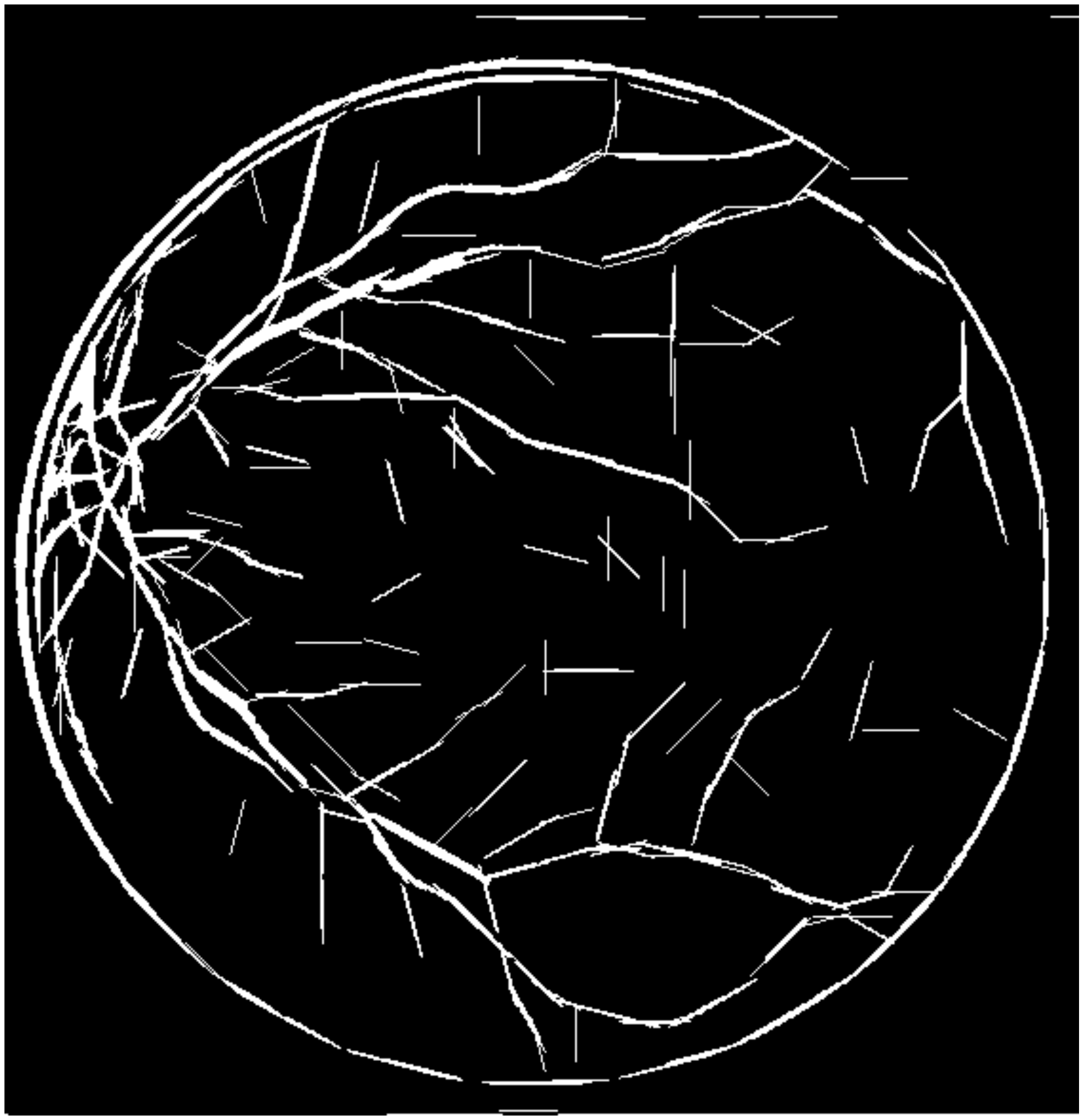
Calculating S_{res}

This step can be summarized in the followings:
1. Opening with 12 structure, r = 2 indicates the structure element 29 pixels long, 1 pixel wide on S_1.
2. Pick the maximum from these 12 images pixel-wise and form an image.
3. If the pixel value ≥ 1, assign 1 on that pixel, otherwise assign 0.

$$S_{\text{res}} = \left(\text{Max}_{i=1..\ \ 12}\left\{\gamma^2_{L_i}(S_2)\right\} \geq 1\right)$$

```
openingOnS2 = Opening[S2, #] & /@ kernels2

openingOnS2Data = ImageData /@ openingOnS2

SresData = MapThread[Max, openingOnS2Data, 2];

sres = Image[SresData]

sresByte = ImageData[sres, "Byte"]

convertToBinary[x_] := If[x ≥ 2, 1, 0]

result = (convertToBinary /@ sresByte[[#]]) & /@ Range[Length[sresByte]]
```

```
Image[result]
```



Future Direction

```
In[23]:= Hyperlink["http://research.google.com/teams/brain/healthcare/"]

Out[23]= http://research.google.com/teams/brain/healthcare/
```

1. For glaucoma detection, there is one approach from a paper which extract the optic disk first, then replace vessels, because vessels are regarded as noises. I will implement this paper as well.
2. Google is working on similar things `http://research.google.com/teams/brain/healthcare/`
3. I think the most important thing is what theory I learned from this paper. I built an App this semester doing segmentation on Corneal Ulcer eyes. But the algorithm I developed is to the binarize on a threshold first and do morphology operations on binary images. Which works

well for the fluorescent images but not working well on images under natural lights. I want to try using the morphology I learned from this paper to solve the natural light problem. There are three medical school students promised me to give me images for testing.