Image Segmentation with Morphology Report

Anthony AOUN Olivier LAURENT

May 9, 2021

Abstract

You may find the code developed for the project of MI206: Morphology and geometric algorithmic on the following public GitHub page :

https://github.com/oZ-0/morphology-IOSTAR.

The algorithms have been developed using oriented-object programming when needed and contain docstrings. A small documentation can be found in the README.md file.

Contents

1 Problem Description

1.1 Context

The main goal of this project is to use image segmentation in order to automatically pinpoint the blood vessels of the eye's retina.

Our target is to apply successive image processing methods to get as close as possible to the manual annotation.

This image segmentation with a morphology approach project has many concrete and beneficial applications. It can, for instance, help detect high blood pressure and kidney failure, knowing that hypertension kills over 10 million people each year.

1.2 Data

We have been given an SLO gray-scale dataset of 10 retina images, of 512×512 pixels weighting around 30kB along with ground-truth, manual annotations of the targeted vessels.

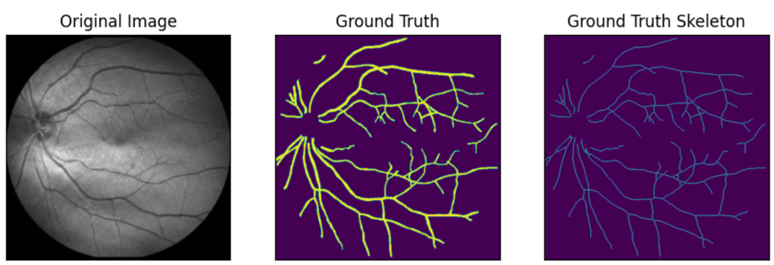


FIGURE 1: Example of a retina image with its expert annotations

1.3 Evaluation metrics

An evaluation function has been written to assess the quality of our segmentation algorithm. This function can be found the in the utils.py module.

The segmentation function outputs an image. The evaluation function consists in comparing the morphological skeletons of the output and the ground truth images. In order to allow slight differences, the computation of True Positives (TP), False Positives (FP) and False Negatives (FN) is relaxed using a 15-steps-morphological thinning of the two images. Three metrics are then computed with the following formulae, Acc being the accuracy and Rec the Recall :

$$Acc = \frac{TP}{TP + FP}$$

$$Rec = \frac{TP}{TP + FN}$$

$$F1 = \frac{2}{\frac{1}{Acc} + \frac{1}{Rec}}$$

Our goal is to maximize F1 and make it as close as possible to 1. Note that the optimization function minimizes -F1.

In order not to overfit the hyper-parameters to a single image, we have maximized the mean of the F1 scores for each image represented by $\overline{F1}$.

2 First strategies

In this first section, we explain quickly our first strategies and detail their results. These strategies are in our GitHub repository but can be sent if you are interested.

2.1 Thresholding

First, as we can see on FIGURE 1, the vessels seem darker than the background. The first and most obvious strategy is thus to use a threshold. This allows to select the darkest parts of the image. There is also a mask to avoid segmenting the image around the retina.

A brute-force functions tries every possible threshold from 50 to 250 and returns the FIGURE 2a. With this threshold, $\overline{F1} = 0.29$. The image 01 OSC segmented by this strategy is on FIGURE 2b on page 4.

2.2 Thresholding and Closing

The first step is to select the angles on the input image. We do this with a Sobel filter.

The second step is to apply a hysteresis threshold instead of a normal threshold. It needs two parameters, the high threshold T_+ and the low threshold T_L. It keeps every pixel higher than T_+ and the connex items linked to these pixels with a value higher than T_-.

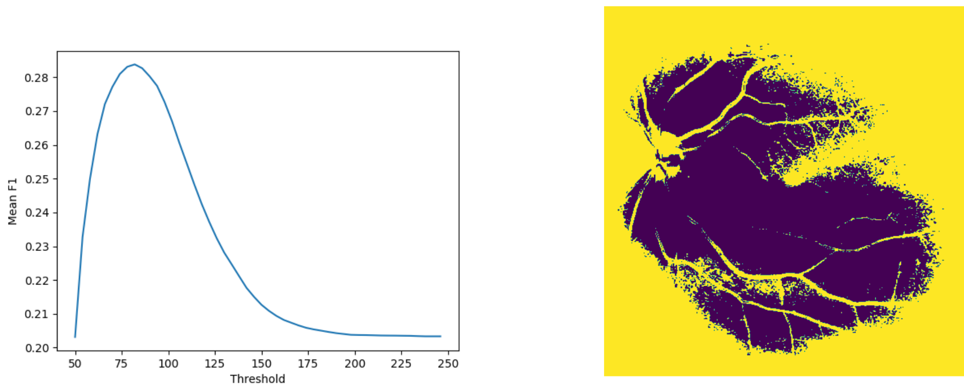
We then apply a generalized closing, constituted of a dilation with a disk of radius 3 and an erosion with a disk of radius 1. The dilation fills the veins and the erosion removes noise and thin the veins.

With a new brute force optimization function on T_+ and T_-, we obtain a best score of $\overline{F1} = 0.821$ for $T_+ = 253$ and $T_- = 243$. The FIGURE 3 is the segmented image of 01 OSC with this set of parameters. One way to easily improve the results would be to remove the outer circle, which creates false positives. This is done in the next strategy.

The results seem a bit too good for the overall quality of the segmentation. The evaluation function might be a little bit too generous.

2.3 First Conclusions

In this section, we saw that obvious transformations, even using morphology and complex filters are not sufficient. The next pages develop a more complex and much more accurate strategy.



(A) $\overline{F1}$ vs. the threshold

(B) The Segmented Image

FIGURE 2: The First Segmentation Strategy

FIGURE 3: The Segmented Image with the second Strategy

3 Main Algorithm

3.1 The Paper

Our first experimentation led to acceptable results. Yet, we wanted to improve the accuracy of our algorithms. To do so, we made a small review of relevant papers. We noticed the paper [3]. This paper gives a detailed explanation of a morphological and differential solution for the segmentation of vessel-like patterns.

3.2 Basic Properties

First of all, let's start by defining the basic properties of our target vessels to better understand the problem :

- **Connectivity :** The vessels form a connected entity that looks like a tree.
- **Shape :** Every vessel is not only characterized by its tall and thin shape, but also by its curve that generally looks like a Gaussian.
- **Position :** Diverge one from the other.
- **Direction :** The vessels globally have very random directions. However, we can narrow their directions' range by assuming that the main directions are represented by the angles going from 0° to 165° with a 15° step.

On a second hand, we should also define the undesirable patterns :

- **Noise :** Caused by undesired elements in the eye's retina or by the digitization of the image.
- **Undesired Areas :** Very dark or bright areas
- **Undesired Shapes :** Especially the ones that look a lot like the thin ramifications of the vessels.

3.3 Our Version of the algorithm

In this section, we propose to explain our algorithm. For each step, please refer to the associated figure for visualization on page 7. The original image, 01 OSC, is on FIGURE 4a.

To get started, we increase the contrast of the image to better distinguish the vessels from the unwanted information. This operation needs a parameter to define the contrast enhancement. At this step, we obtain FIGURE 4b.

The following step is to detect the vessels by generating linear structural elements that can be very precisely defined by picking their length and orientation (utils.py). These structural elements are very adapted to our model, not only for their linear form, but also for their various directions that cover the majority of the vessels. Using an algebraic opening, we reinforce the contrast of dark structures where our elements fit. This step corresponds to the FIGURE 4c.

The next step is applying the top-hat transformation. We apply the conjugate top-hat transformation for each linear structuring element on the processed image in order to select each time a connected group of vessels that have the same direction as this structural element. After that, we sum all the top-hats to obtain the fully constructed aggregation of vessels. This step also helps remove the outside circle that surrounds the vessels' aggregation. It is crucial to mention that this operation requires accurate structural elements that aren't too long (to get the small ramifications of vessels) and aren't too short (to get the minimum amount of noise). Indeed, on of the biggest challenges in this project is to find the optimum quantity of information that we must remove; removing noise without affecting the vessels. At this step, we obtain FIGURE 5a.

The first attempt to clean the remaining noise is to use one of the most common morphology transformation : the algebraic erosion. This transformation not only cleans a part of the noise, but also helps cleaning the central nucleus that we have to remove. However, by doing this, we also removed a part of the essential information : thin vessel ramifications. This can be seen on FIGURE 5b. We had to find a way to bring these ramifications back.

To bring back the information, we apply a geodesic reconstruction (by dilation) with the image of the previous step. This operation brings back some of the essential information but also noise with it as you can see on FIGURE 5c. It must be followed by a thorough cleaning process.

Our final cleaning process thus consists in applying a Gaussian filter parametered by σ. We obtain the FIGURE 6a.

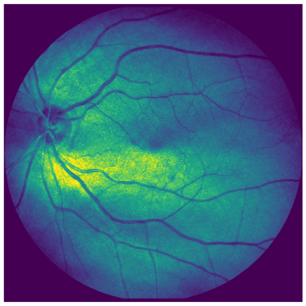
We then apply a hysteresis threshold to keep as many as possible connex items. We remove the outer circle and are good to go. The final image is FIGURE 6b.

3.4 Hyper-parameters optimization

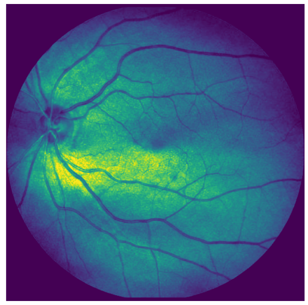
At first, we developed functions that you can find in the optimizers module to find the best parameters. The first naive methods quickly started to become slow. There are indeed 7 hyper-parameters in our final segmentation function.

- The low and high thresholds for the hysteresis thresholding
- The contrast enhancement value
- The lengths of the linear structuring elements for the 3 morphological transformations
- The standard deviation of the final Gaussian filter

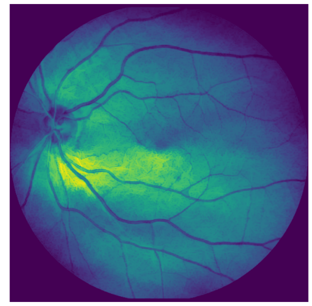
To be more efficient, we decided to use the package optuna [1]. It uses multiprocessing to test several hyper-parameters at the same time and searches the optimum using the Tree-Structured Parzen algorithm [2].



(A) The Original image

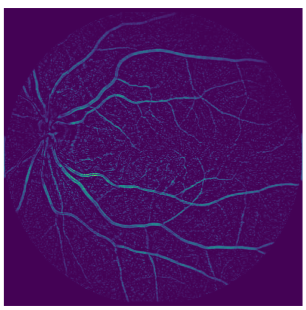


(B) Contrast Enhancement

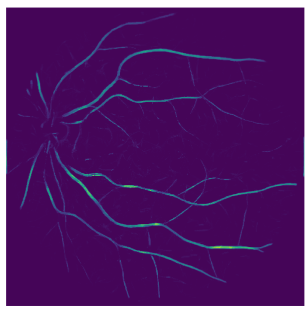


(C) Algebraic Opening

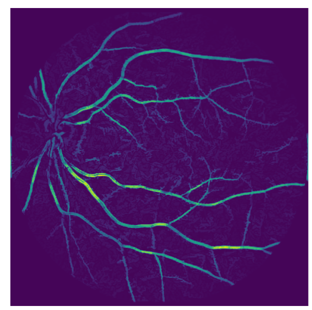
FIGURE 4: Reinforcing contrast



(A) Conjugate Top-hats

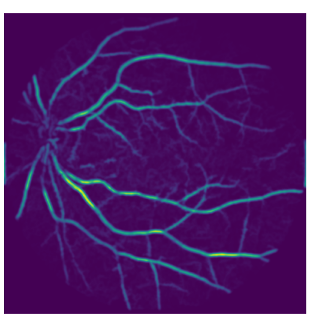


(B) Erosion

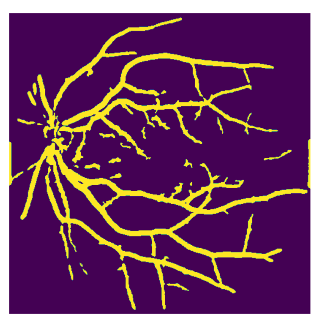


(C) Reconstruction

FIGURE 5: Exploiting connex items



(A) Gaussian filter



(B) Final Image

FIGURE 6: Filtering and Thresholding

4 Results

After running optuna for a few thousand epochs refining the ranges, we obtained the following results. The summaries for 2 images, 21 OSC and 48 OSN are showed on FIGURE 7 and FIGURE 8 on page 9.

4.1 Best Parameters

Parameter	Best value
Low Threshold	48.7376
High Threshold	64.0953
Contrast enhancement	1.73199
Length of structures 1	15
Length of structures 2	7
Length of structures 3	15
STD of the Gaussian filter	1.558186

4.2 Table of Results

	F1 (%)	Accuracy (%)	Recall (%)
Mean	95.173	96.384	94.030
01 OSC	95.779	96.461	95.108
02 OSC	95.474	93.662	97.359
03 OSN	95.548	98.019	93.200
08 OSN	96.221	98.489	94.054
21 OSC	93.455	96.504	90.593
26 ODC	94.372	96.800	92.062
28 ODN	94.206	95.623	92.831
32 ODC	94.458	94.601	94.315
37 ODN	95.582	95.961	95.207
48 OSN	96.632	97.719	95.568

4.3 Sources of Error

To find the sources of error, we modified the proposed version of the evaluation function. It now returns the False positives and False negatives maps. These maps can be showed if the *do_map* parameter is set to True.

The first image, the FIGURE 9 on page 10, is the worst among our set of parameters. We see that False positives consist mainly of noise. False negatives are very important vessels which have been confused with noise by our algorithm.

On the other hand, the FIGURE 10, the second image's false positives correspond mainly to the central nucleus. Concerning false negatives, some vessels have been forgotten away from the center. It seems that a radial threshold could help in this situation, or perhaps a radial pre-processing.

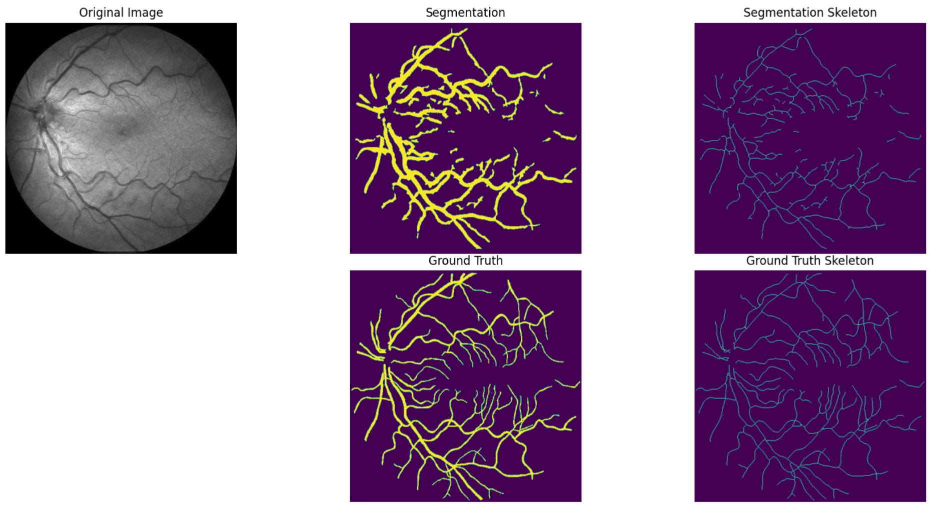


FIGURE 7: Summaries of 21 OSC - Worst result

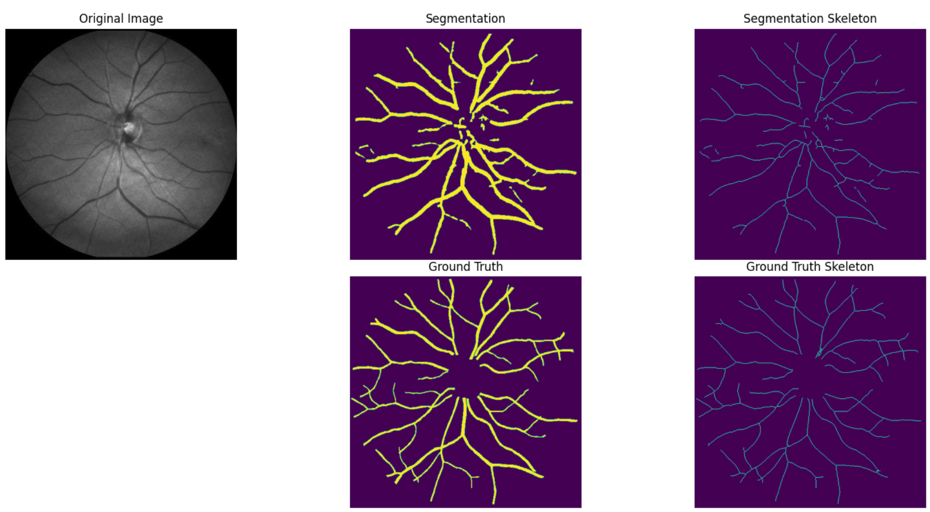


FIGURE 8: Summaries of 48 OSN - Best result

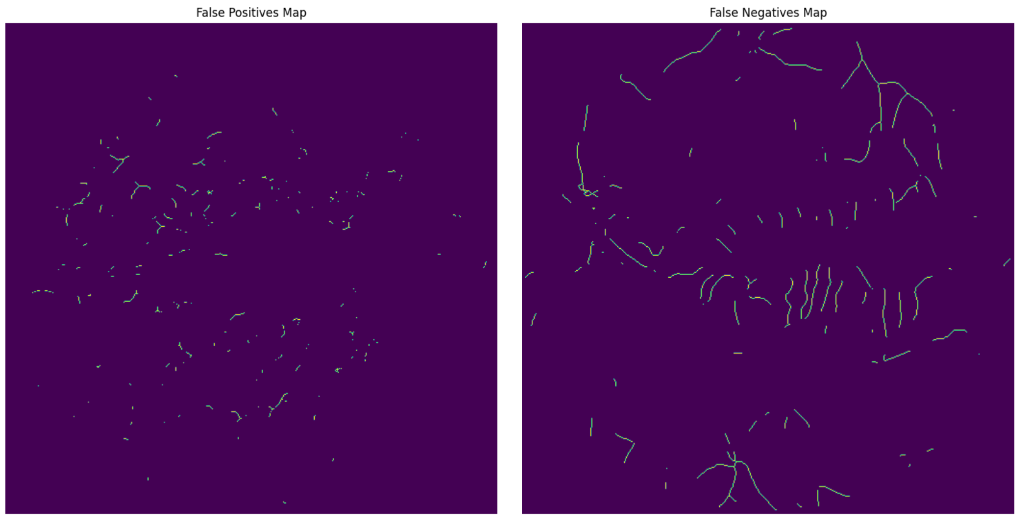


FIGURE 9: Error maps of 21 OSC - Worst result

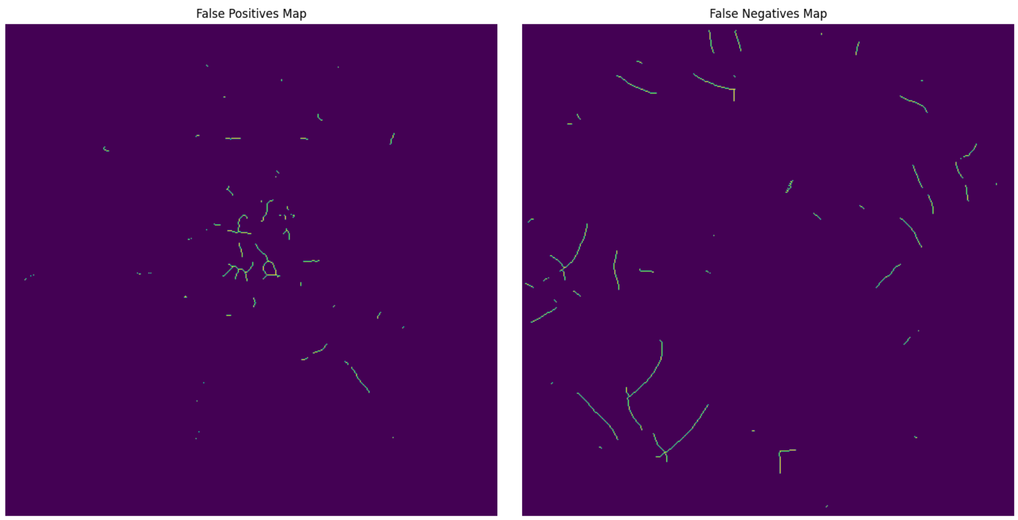


FIGURE 10: Error maps of 48 OSN - Best result

5 Going Further

As we have seen in the previous section, an important part of the false positives error comes from the central nucleus. This central nucleus is very difficult to isolate from the rest of the image. It is indeed a very complex structure. The luminosity varies in its center but also between the images. More reseach is needed to find a satisfactory algorithm.

In the previous part, one could see that some vessels are forgotten. This can happen because of a non sufficient contrast for instance. More morphological operations could help to avoid losing information.

We could also add a first preprocessing step to ensure that all images can be compared. This step is not obvious since the images vary a lot.

The threshold could evolve radially (reduced or increased depending on the luminosity).

On a general basis, it could be interesting to have a bigger database. This could reduce the overfitting of our hyper-parameters on the data. We could also separate the database in validation and test sets.

In this experiment, we have used morphology as well as differential operators. One last and very open idea could be to complete this methods with Neural networks or Markov chains and thus have better results.

6 Conclusions

In this report, we have seen that morphological operators were totally suitable for the task of segmenting vessel-like patterns. While we used differential operators to complete our algorithm, it is highly likely that morphological operators could have helped for contrast enhancing as well as filtering requiring perhaps some more work.

The results obtained with these relaxed evaluation criteria seem satisfactory, even though not optimal. We could wonder whether these criteria as well as the results are rigorous enough for real life applications.

References

[1] Takuya Akiba et al. “Optuna: A Next-generation Hyperparameter Optimization Framework”. In: *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2019.

[2] James Bergstra et al. “Algorithms for Hyper-Parameter Optimization”. In: *25th Annual Conference on Neural Information Processing Systems (NIPS 2011)*. Ed. by J. Shawe-Taylor et al. Vol. 24. Advances in Neural Information Processing Systems. Granada, Spain: Neural Information Processing Systems Foundation, Dec. 2011. URL: https://hal.inria.fr/hal-00642998.

[3] F. Zana and J.-C. Klein. “Segmentation of vessel-like patterns using mathematical morphology and curvature evaluation”. In: *IEEE Transactions on Image Processing* 10.7 (2001), pp. 1010–1019. DOI: 10.1109/83.931095.