IDRiD Sub-Challenge 3:
Optic Disc and Fovea Detection

Natalie Tipton

Grand Valley State University
EGR 532
Dr. Samhita Rhodes

4/20/2019

Abstract

Diabetic retinopathy is a disease of the eye caused by prolonged elevated blood glucose levels. People suffering from diabetes should have retinal scans annually to check for this as symptoms are often non-existent until the loss of vision. Currently these scans are analyzed manually by a trained professional which is time consuming and leaves room for human error. This algorithm automated the beginning process of this analysis by detecting the center point of the two most important landmarks when assessing diabetic retinopathy: the optic disc and the fovea. This was done through a series of contrast stretching, smoothing filters, thresholding, binarization, masking, and neighborhood analysis. The algorithm led to coordinates of the center of the optic disc with a percent error less than ten over 96 percent of the time and coordinates of the center of the fovea with a percent error less than ten over 87 percent of the time. While these results prove that the algorithm was relatively successful, there is still room for improvement. Including blood vessel behavior into the algorithm could reduce the error for both landmarks. Incorporation of an algorithm such as this into common medical practice could allow for diabetics to receive retinal scans more frequently, reducing the commonality of blindness due to diabetic retinopathy.

Introduction

Diabetes mellitus is a chronic illness characterized by hyperglycemia due to the body's inability to create or absorb insulin [7]. A prolonged increase in blood sugar can cause those suffering from diabetes to be more susceptible to developing diabetic retinopathy, the most common cause of avoidable vision impairment [3]. This is due to leakage of blood and blood constituents over the retina. If gone undetected, this disease can cause blindness in those affected. However, if it is caught early, it can be treated and reversed [5]. Because of this, it is suggested that patients with diabetes undergo an annual retinal exam to check for signs of the presence of diabetic retinopathy so that it can be caught and reversed before severe damage is done.

Currently, diabetic retinal scans are completed in an ophthalmologist's office. The image is taken and the scans are analyzed manually by the ophthalmologist or another professional trained to read the scans [4]. The first step in this analysis is to determine where the region of interest for them to examine is. This region of interest is determined by the optic disc (OD) and the fovea. Segmenting the OD reduces confusion when looking for bright lesions on the retina. The fovea is the center of one's vision and the OD is where the optic nerve enters the retina. Bright lesions or dark hemorrhages are most indicative of severe issues related to diabetic retinopathy when located within a certain distance from the fovea. Damage here is what can lead to macular oedema, which causes blindness. Therefore, when completing diabetic retinal scans, the most important place to look for signs of disease is within a circular area around the center of the fovea with a diameter equal to the diameter of the OD [7]. As a result, during manual assessment of a retinal scan, size of the OD and the exact center point of the fovea must be determined in order to ensure the correct location in the retina is being examined. This process requires the reader to interpret different brightnesses in the image as they appear to the naked eye. The reader must use their best judgement to determine where the center location of the OD is and where its outer limits lie. Based upon that information, they must determine the center of the fovea and where a circle with the diameter of the OD should be placed to limit the region of interest. This may involve the use of a ruler or another similar measuring device that has limited precision. There is much room for human error in this process. If the size of the OD and the location of the fovea are not determined correctly, the reader may look in too large of a region and detect artifacts that do not affect the fovea or too small of a region and miss artifacts that do. The process is also very time consuming and requires trained professionals to perform it.

The development of the algorithm described in this paper can be incredibly advantageous for patients of diabetes as well as ophthalmologists. As the system requires no user input, it eliminates the chance for human error. It also eliminates much of the time required to complete analysis. If this algorithm was incorporated with an automated diagnosis scheme, it could play a huge part in decreasing the number of people affected with vision loss from diabetic retinopathy. Scans could be completed in primary care offices as opposed to specialized ophthalmologist offices [4], allowing increased access to scans for people suffering from diabetes, particularly those who were limited by insurance or simply did not know they needed to go to the eye doctor to get the tests done.

Aims

Before creating the algorithm, a plan of action was developed for how to begin finding a solution to the issue of time consuming, error prone retinal scan analysis. An algorithm would be created to allow for an automated system that requires no input from the user aside from the location of the folder containing the image(s) to be assessed. The output of this system would be in the form of an image and of data exported to a Microsoft Excel sheet. The image would allow the user to visualize where exactly the OD was by showing pure white over the location of the entire OD and the original color image everywhere else. It would also should them where the center point of the OD and fovea is with a mark at the center coordinates. The data exported to an Excel sheet would show exact coordinates for both the fovea and the OD. When planning the flow of the algorithm, segmentation of the OD was first priority. This is because localization of the center point of the OD would be found easily if segmentation was completed first and the location of the fovea is related to the location of the OD, so that would come last.

As the optic disc is generally the brightest location in the fundus image and the fovea is generally the darkest location, contrast was a very important feature to enhance. Increasing contrast would allow for an easier time with OD segmentation. Therefore, the original images would be split up into different planes including the red, blue, and green planes of the RGB version of the image and the hue, saturation, and intensity planes of the HSI version. These would be analyzed to see which one shows the starkest difference between the bright OD and the dark fovea. Once that was found, it would be contrast stretched to obtain the maximum difference between dark and bright spots on the image. This would be tested with multiple different contrast stretching techniques to determine which would be most useful. From there, the image could be binarized based upon a determined threshold to mask out everything except the brightest pixels, which should signify the OD. Smoothing should be used if necessary to make this action more specific and avoid detecting anomalous bright spots. This process may be repeated until an accurate segmentation of the OD is complete. Using this information, MATLAB's toolkit could be utilized to find the centroid of the region created in the mask in order to determine the location of the center of the OD.

The localization of the fovea may be more difficult since there are more dark spots in the image than just the fovea. This could make detection of it based solely on its intensity difficult. Research showed that the fovea typically lies within a radius of two to two and a half times the diameter of the OD away from the OD [6]. Analysis of the data given corroborated that, so determining the location of the fovea based upon the location of the OD seems to be the most reasonable route to take. The original image will be masked to show only the location the fovea is expected to lie within and binarization will be completed. Once again, smoothing may be necessary to obtain an accurate region. The MATLAB tool to find the centroid of a region will be implemented to try to obtain coordinates to the center point.

Methods

The images used for the training and testing of this algorithm were fundus images captured by a retinal specialist in India using a Kowa VX-10 alha digital fundus camera with a 50 degree field of view. 516 images in total were provided and all are centered near the macula. The resolution of the images was 4288x2848 pixels, stored in a .jpg file format [3]. Along with images, groundtruth values were also provided in a .csv file to display the coordinates of the center points for the OD and the fovea for each image. This data would be used to verify algorithm results.

The algorithm came together relatively similarly to what was expected in the original plan of action detailed in the specific aims section. There were, however, certain discoveries that helped improve the original plan. The first step was completing OD segmentation. In order to do this, the plane of the original image that showed the most contrast between the optic disc and the fovea was chosen. Between red, green, blue, hue, saturation, and intensity, the red plane wielded the best results.

In order to optimize contrast, the red plane then underwent histogram equalization. This spread out the intensities present in the image so that each intensity from 0 to 255 were represented relatively equally in the image. This is an effective way to increase contrast in an image. The next step was to determine what part of the image was the OD by implementing thresholding and binarization. First, however, a 50x50 neighborhood smoothing filter was implemented. This performed averaging on each neighborhood and assigned the center pixel to the average of the entire neighborhood. This is useful before binarization because accuracy was very important. If there were small regions of pixels that were brighter than the rest around them, smoothing the image would have toned them down to better match what was surrounding them. Doing this resulted in less pixels binarized as true for being part of the OD when they were not connected.

Once smoothing was complete, a threshold was set at 99 percent of the maximum intensity. This threshold value was obtained through trial and error. Since histogram equalization set the OD at an intensity greater than the rest of the image, this threshold worked relatively well. It was not perfect, however, and the binary image still showed additional white pixels that did not belong to the OD if relatively bright lesions were present in the retinal image. Because of this, another smoothing filter was applied. This time, it was completed with a neighborhood size of 10x10. This neighborhood size was good to eliminate individual or small groups of pixels that were incorrectly classified while not over-smoothing and eliminating pixels that were correct. This smoothed image was then re-binarized to eliminate the gray intensities that were created by smoothing. This was done with a threshold of 90 percent. That threshold value was determined by trial and error as it was the one that resulted in the correct classification.

After smoothing was complete, the OD mask was closer to the accurate result that was expected, however, blood vessels entering the OD tended to cause missing sections of the circular OD. In order to account for that, the MATLAB function imclose was used with a disk size of 100. This meant that any region containing an open space that could be closed by a disk with a radius of 100 pixels was closed. This allowed for the mask of the OD to be closed up and appear more circular to fit the shape of the actual landmark.

To finalize the creation of the OD mask and finish segmentation, one last step was necessary. Despite multiple smoothing efforts, some images still resulted in binary regions that were not the OD. In order to remove them, the MATLAB function bwareafilt was used. This function looked for 8x8 connected regions and found the area of them. It then removed all regions aside from the one with the largest area. This allowed for any erroneous bright pixels to be removed, leaving only the OD. That concluded the segmentation portion of this algorithm.

After the OD was segmented, the center point was able to be determined. This was done using another MATLAB function, regionprops. This function looks for 8x8 connected neighborhoods, similarly to bwareafilt. It has additional capabilities, however, to determine the centroid of the regions if specified in the function call. It returned an (x,y) coordinate to this center location that could be retrieved and exported.

Once segmentation and localization of the OD was finished, the portion of this algorithm was to detect the fovea and its center location. At first, thresholding and binarization was attempted similarly to the segmentation of the OD except for dark pixels instead of bright. However, the image contained too many dark areas for this to be successful. The outer border of the retinal scan was often as dark or darker than the fovea. After analyzing the ground truth values that were given for the centers of the OD and fovea, it became clear that the location of the fovea was dependent on the location of the OD. In all cases, the fovea was located within 1200 and 1500 pixels to the side of the OD and 500 pixels above or below. This knowledge allowed for a rectangular mask to be created to limit the area that was to be thresholded down to a specific section of the eye that the fovea was known to exist within. This rectangular mask was multiplied by the grayscale version of the original image to obtain a section of the eye to further analyze. Once this was done, the same process that was used to create the OD mask was used to create a binary region over the location of the fovea. First, the masked image was binarized where any pixels less than 110 percent of the minimum value, but not completely 0, was turned white and everything else was turned black. This resulted in the general form of the fovea shape, however, it was not complete. The shape consisted of only a few pixels here and there. Because of this, a 3x3 smoothing filter was applied. This allowed for the shape to be filled in more by creating lighter intensity gray pixels in the shape of the fovea. This smoothed result was thresholded where any values greater than 10 percent of the maximum intensity were turned white. This threshold was chosen because the intent was to turn pixels that were black before into white. By making it a small percentage of the maximum intensity allows for pixels that were black and were increased by being averaged with a white pixel will be classified as true. After this, the imclose function was used again with a disk size of 25. This size was chosen because the region of the fovea is relatively small. Choosing a radius of 25 allowed for the fovea to be filled in without overextending the region. The function bwareafilt was applied again to remove any regions aside from the largest one present in the case that there were still white pixels that were not a part of the fovea. Since segmentation of the fovea was not important, this mask was only used to create a region that could be assessed for its centroid. The regionprops function was used to return the centroid of the fovea.

In order to obtain the results of this algorithm, the output was plotted as well as exported to Microsoft Excel. Each step mentioned above was plotted onto an image so that the progress of the algorithm could be followed. These images are not necessary for the functionality and could be commented out to save time. The more important result format is the exportation to Excel. For this, centroid values for both landmarks were rounded to the nearest whole number to match the format of the given ground truths to make comparison easier. The centroid coordinates were then exported to an excel sheets in columns next to one another.

This algorithm was trained using a all of the 516 images that were available from the IDRiD website. Different sets of images were run through at a time to see how each of them performed while creating the algorithm so that a variety of scenarios could be accounted for. This allowed for changes to be made to optimize the code that would not have been noticed had only a few images been used to test with. Once the algorithm was successful based upon qualitative visual inspection of the training data outputs, the given test images were used to verify the results. The 103 given test images were run through the algorithm at once. All images except the final image were suppressed to save time. The coordinate data was exported to the Excel sheet that was provided and contained the ground truth values. This allowed for Excel calculations to be performed to obtain different metrics that could allow the quantitative success of the algorithm to be determined. The difference between ground truth and resulting data, the percent error between them, the number of images that had an error greater than 10 percent, and the percent of total images with an error greater than 10 percent were all found through Excel. 10 percent was chosen as the threshold error value as this is typically considered a reasonable amount of error in engineering.

Results

The results from each step described above were generally successful. To begin, the algorithm was able to handle as many images in a selected directory as desired. For much of the training time, one image would be run in the directory. However, at one time, all 413 training images were run together. While this caused the runtime to increase dramatically, the algorithm was able to handle them all and eventually give an output.

Segmentation of the OD was quite successful. Intermediate masking steps can be seen in Appendix A, Figures x through x. These steps will show how filtering and masking were applied to create the final segmented region. One of the most important parts of these beginning steps was the histogram equalization that occurred on the red plane. This allowed for contrast to be increased enough to the algorithm to determine the difference between locations. An example transformation curve from this process is shown below.

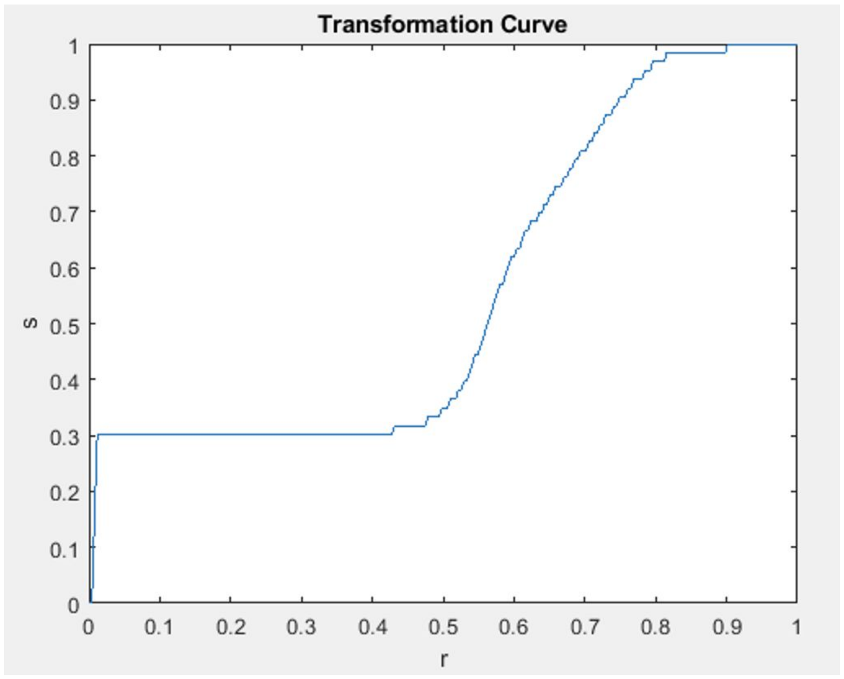

Figure 1. Transformation curve for histogram equalization

According to the Excel data shown in Table 1, OD localization was successful 96.12 percent of the time. A final view of the segmentation and localization of the OD can be found in Figures 2 and 3. These figures show a white circular region over where the OD is and a blue star at the coordinates of the two center points. Notice that the algorithm was able to detect the OD no matter which side of the eye it was located on. The OD was also able to be detected despite the fact that there were other bright spots within the retinal images shown.

Table 1. First 35 OD centroid results compared to ground truth values

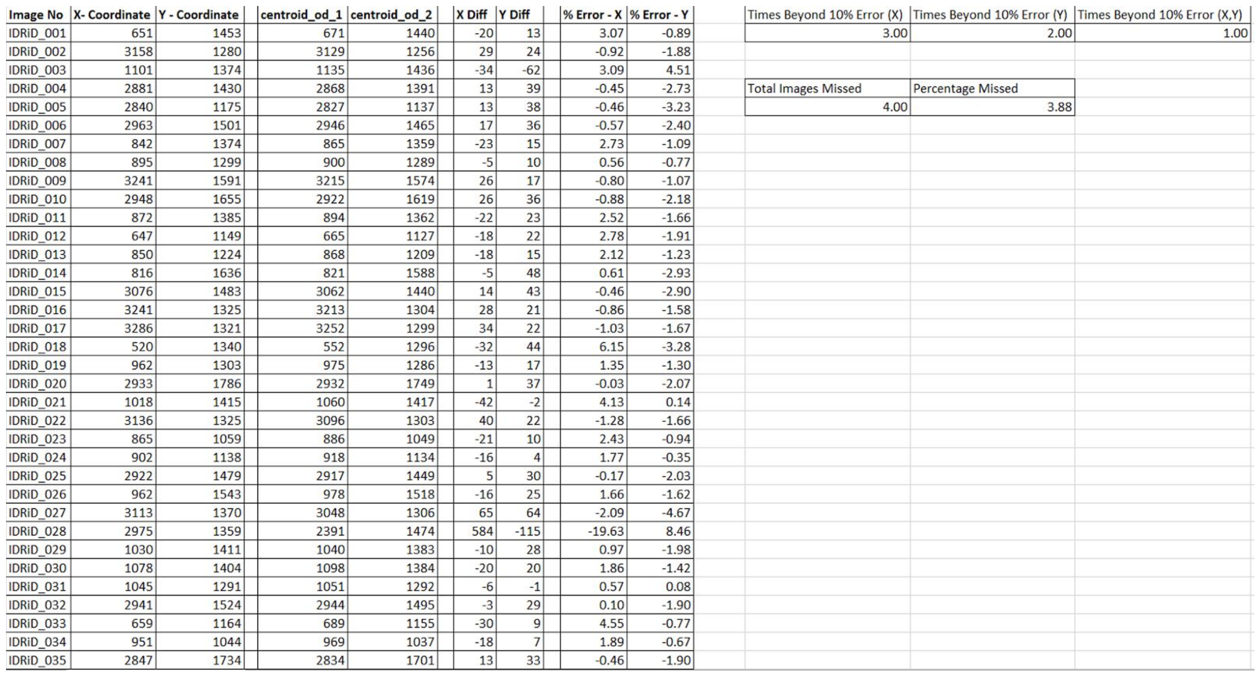

Image No	X- Coordinate	Y - Coordinate	centroid_od_1	centroid_od_2	X Diff	Y Diff	% Error - X	% Error - Y
IDRiD_001	651	1453	671	1440	-20	13	3.07	-0.89
IDRiD_002	3158	1280	3129	1256	29	24	-0.92	-1.88
IDRiD_003	1101	1374	1135	1436	-34	-62	3.09	4.51
IDRiD_004	2881	1430	2868	1391	13	39	-0.45	-2.73
IDRiD_005	2840	1175	2827	1137	13	38	-0.46	-3.23
IDRiD_006	2963	1501	2946	1465	17	36	-0.57	-2.40
IDRiD_007	842	1374	865	1359	-23	15	2.73	-1.09
IDRiD_008	895	1299	900	1289	-5	10	0.56	-0.77
IDRiD_009	3241	1591	3215	1574	26	17	-0.80	-1.07
IDRiD_010	2948	1655	2922	1619	26	36	-0.88	-2.18
IDRiD_011	872	1385	894	1362	-22	23	2.52	-1.66
IDRiD_012	647	1149	665	1127	-18	22	2.78	-1.91
IDRiD_013	850	1224	868	1209	-18	15	2.12	-1.23
IDRiD_014	816	1636	821	1588	-5	48	0.61	-2.93
IDRiD_015	3076	1483	3062	1440	14	43	-0.46	-2.90
IDRiD_016	3241	1325	3213	1304	28	21	-0.86	-1.58
IDRiD_017	3286	1321	3252	1299	34	22	-1.03	-1.67
IDRiD_018	520	1340	552	1296	-32	44	6.15	-3.28
IDRiD_019	962	1303	975	1286	-13	17	1.35	-1.30
IDRiD_020	2933	1786	2932	1749	1	37	-0.03	-2.07
IDRiD_021	1018	1415	1060	1417	-42	-2	4.13	0.14
IDRiD_022	3136	1325	3096	1303	40	22	-1.28	-1.66
IDRiD_023	865	1059	886	1049	-21	10	2.43	-0.94
IDRiD_024	902	1138	918	1134	-16	4	1.77	-0.35
IDRiD_025	2922	1479	2917	1449	5	30	-0.17	-2.03
IDRiD_026	962	1543	978	1518	-16	25	1.66	-1.62
IDRiD_027	3113	1370	3048	1306	65	64	-2.09	-4.67
IDRiD_028	2975	1359	2391	1474	584	-115	-19.63	8.46
IDRiD_029	1030	1411	1040	1383	-10	28	0.97	-1.98
IDRiD_030	1078	1404	1098	1384	-20	20	1.86	-1.42
IDRiD_031	1045	1291	1051	1292	-6	-1	0.57	0.08
IDRiD_032	2941	1524	2944	1495	-3	29	0.10	-1.90
IDRiD_033	659	1164	689	1155	-30	9	4.55	-0.77
IDRiD_034	951	1044	969	1037	-18	7	1.89	-0.67
IDRiD_035	2847	1734	2834	1701	13	33	-0.46	-1.90

Times Beyond 10% Error (X)	Times Beyond 10% Error (Y)	Times Beyond 10% Error (X,Y)
3.00	2.00	1.00

Total Images Missed	Percentage Missed
4.00	3.88

Fovea detection was slightly less successful than OD detection. As the Excel data in Table 2 shows, fovea localization was correct 87.38 percent of the time. This is slightly above 10 percent error, and therefore, leaves room for improvement. The intermediate filtering and binarization steps for this can be found in Appendix A, Figures x through x. The output images in Figures 2 and 3 as mentioned before also show fovea data in addition to the OD segmentation and localization. These images show that fovea detection was successful even when a lot of darkness was present in the image that was not the fovea. By visual inspection of these images, it is slightly obvious that the algorithms ability to very accurately detect the center of the OD is more refined than its ability to detect the exact center of the fovea. In Figure 3, it appears that the fovea center point is slightly above the actual center of the fovea. However, since this difference is very small, the algorithm was still fairly successful.

Table 2. First 35 fovea centroid results compared to ground truth values

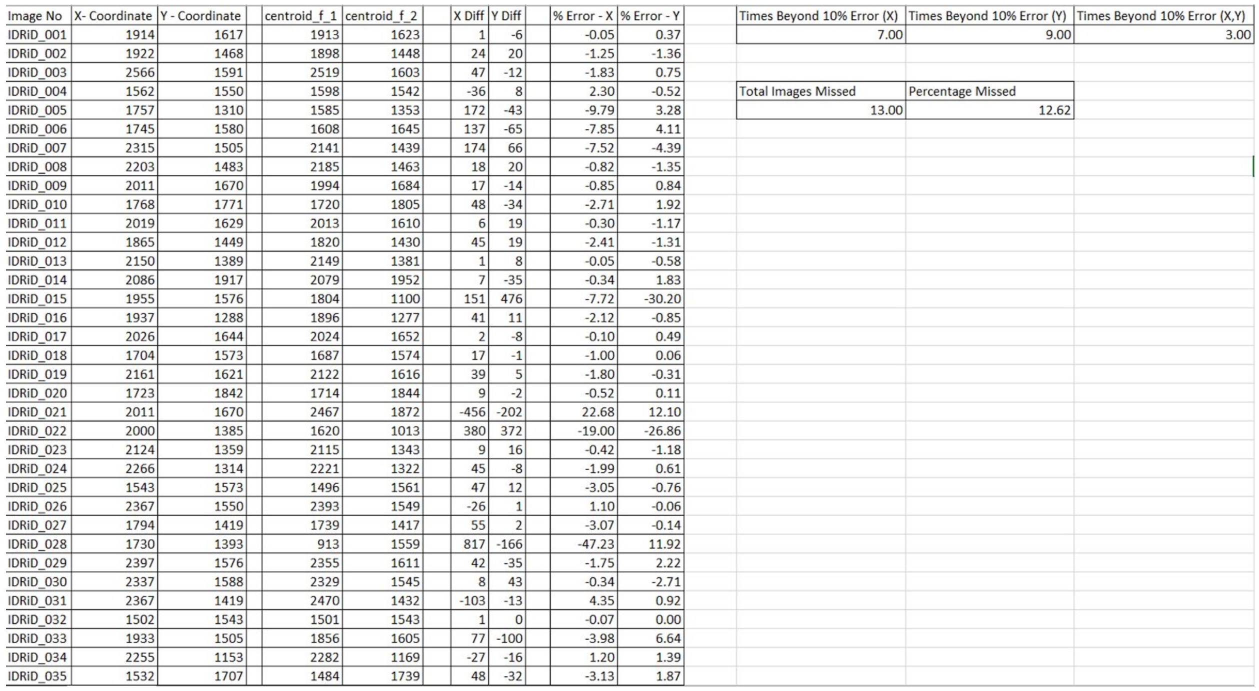

Image No	X- Coordinate	Y - Coordinate	centroid_f_1	centroid_f_2	X Diff	Y Diff	% Error - X	% Error - Y
IDRiD_001	1914	1617	1913	1623	1	-6	-0.05	0.37
IDRiD_002	1922	1468	1898	1448	24	20	-1.25	-1.36
IDRiD_003	2566	1591	2519	1603	47	-12	-1.83	0.75
IDRiD_004	1562	1550	1598	1542	-36	8	2.30	-0.52
IDRiD_005	1757	1310	1585	1353	172	-43	-9.79	3.28
IDRiD_006	1745	1580	1608	1645	137	-65	-7.85	4.11
IDRiD_007	2315	1505	2141	1439	174	66	-7.52	-4.39
IDRiD_008	2203	1483	2185	1463	18	20	-0.82	-1.35
IDRiD_009	2011	1670	1994	1684	17	-14	-0.85	0.84
IDRiD_010	1768	1771	1720	1805	48	-34	-2.71	1.92
IDRiD_011	2019	1629	2013	1610	6	19	-0.30	-1.17
IDRiD_012	1865	1449	1820	1430	45	19	-2.41	-1.31
IDRiD_013	2150	1389	2149	1381	1	8	-0.05	-0.58
IDRiD_014	2086	1917	2079	1952	7	-35	-0.34	1.83
IDRiD_015	1955	1576	1804	1100	151	476	-7.72	-30.20
IDRiD_016	1937	1288	1896	1277	41	11	-2.12	-0.85
IDRiD_017	2026	1644	2024	1652	2	-8	-0.10	0.49
IDRiD_018	1704	1573	1687	1574	17	-1	-1.00	0.06
IDRiD_019	2161	1621	2122	1616	39	5	-1.80	-0.31
IDRiD_020	1723	1842	1714	1844	9	-2	-0.52	0.11
IDRiD_021	2011	1670	2467	1872	-456	-202	22.68	12.10
IDRiD_022	2000	1385	1620	1013	380	372	-19.00	-26.86
IDRiD_023	2124	1359	2115	1343	9	16	-0.42	-1.18
IDRiD_024	2266	1314	2221	1322	45	-8	-1.99	0.61
IDRiD_025	1543	1573	1496	1561	47	12	-3.05	-0.76
IDRiD_026	2367	1550	2393	1549	-26	1	1.10	-0.06
IDRiD_027	1794	1419	1739	1417	55	2	-3.07	-0.14
IDRiD_028	1730	1393	913	1559	817	-166	-47.23	11.92
IDRiD_029	2397	1576	2355	1611	42	-35	-1.75	2.22
IDRiD_030	2337	1588	2329	1545	8	43	-0.34	-2.71
IDRiD_031	2367	1419	2470	1432	-103	-13	4.35	0.92
IDRiD_032	1502	1543	1501	1543	1	0	-0.07	0.00
IDRiD_033	1933	1505	1856	1605	77	-100	-3.98	6.64
IDRiD_034	2255	1153	2282	1169	-27	-16	1.20	1.39
IDRiD_035	1532	1707	1484	1739	48	-32	-3.13	1.87

Times Beyond 10% Error (X)	Times Beyond 10% Error (Y)	Times Beyond 10% Error (X,Y)
7.00	9.00	3.00

Total Images Missed	Percentage Missed
13.00	12.62

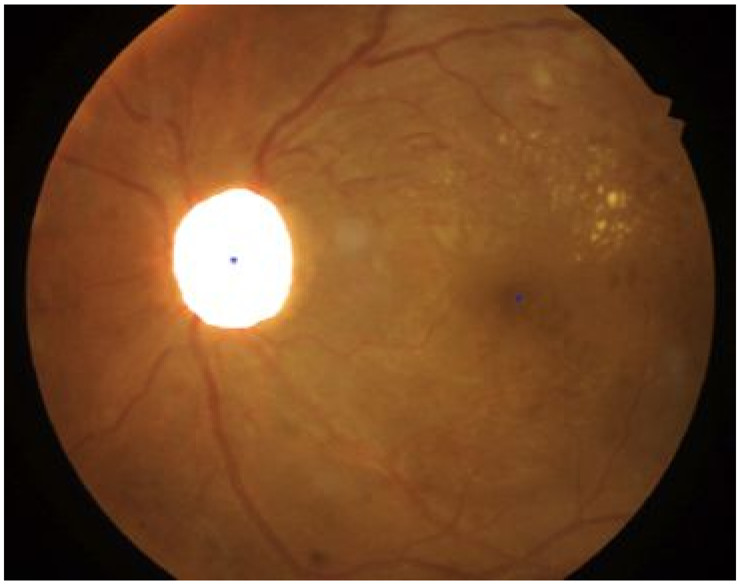
Figure 2. Output image 1

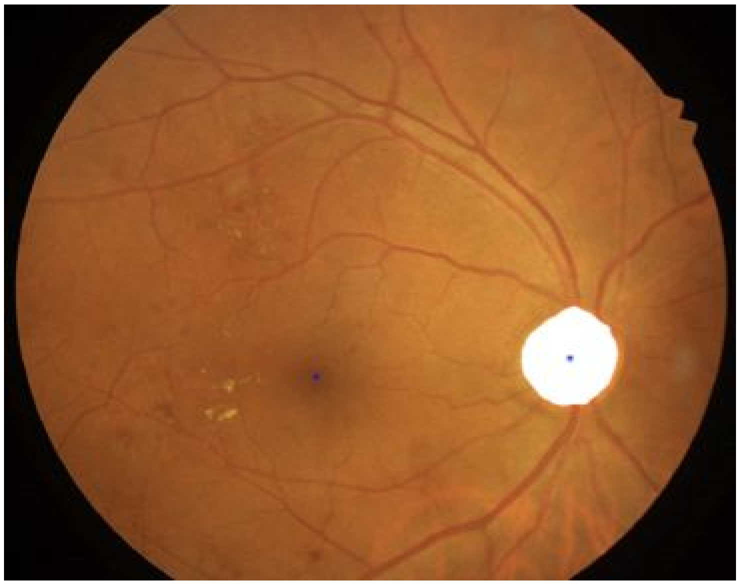
Figure 3. Output image 2

Discussion

This algorithm was successful in many aspects. OD segmentation and localization was performed very well, and results lie easily within the acceptable range of error with only 3.88 percent error. This worked so well because of the high contrast that was able to be obtained using the red plane of the image combined with histogram equalization. As long as the image did not contain a very bright spot that was larger than the OD, the algorithm was successful. While this did occur in a few of the images tested, it was not a common occurrence. This type of image would be indicative of severe disease in the eye, and a failed test through this algorithm would tell the technician running the test that something is wrong. Therefore, even though it failed based upon what this algorithm is supposed to do, it is still providing some meaningful result.

The detection of the fovea had an higher error than the OD detection, which makes sense since its analysis was all based off of the location of the OD. That means that if the OD showed error, it was likely that the fovea would show error as well. This added to the small error that is to be expected with any data to begin with. If more contrast could have been obtained between the fovea and the background of the rest of the eye, fovea localization could have been more successful. The lack of contrast here is because of the fact that the fovea was not completely black. It was a darker gray intensity than the rest of the retina, but not by a drastic amount. The OD showed a very significant difference in intensity when compared to the background gray intensity of the eye. This made detection there very easy. If a similar amount of contrast was present with the fovea as well, results could have been improved.

While overall, the algorithm was successful, there is still room for improvement. Ideally, all error would be reduced to below 10 percent across the board. In order to do this, fovea detection would need to be improved. One piece of information learned from research could be especially helpful for this. The blood vessels in the retina all converge at the OD and the fovea is entirely devoid of them [6]. Therefore, if the vessel tree could be determined, detection of the two landmarks would be much easier and more accurate. This could be particularly helpful for detection of the fovea as present of a blood vessel could immediately inform the algorithm that the fovea is not located there and it could reassess. Another potential room for improvement would be to explore different filters that could further increase the contrast of the image. Other additions that could be made include utilizing filters such as a median filter to remove noise in the segmentation of the OD and fovea. This could help get an accurate OD segmentation from the start, increasing the likelihood that the fovea detection would also be accurate.

References

[1] Prasanna Porwal, Samiksha Pachade, Ravi Kamble, Manesh Kokare, Girish Deshmukh, Vivek Sahasrabuddhe and Fabrice Meriaudeau, "Indian Diabetic Retinopathy Image Dataset (IDRiD)", IEEE Dataport, 2018. [Online]. Available: http://dx.doi.org/10.21227/H25W98.

[2] Porwal P, Pachade S, Kamble R, Kokare M, Deshmukh G, Sahasrabuddhe V, Meriaudeau F. Indian Diabetic Retinopathy Image Dataset (IDRiD): A Database for Diabetic Retinopathy Screening Research. Data. 2018; 3(3):25. Available (Open Access): http://www.mdpi.com/2306-5729/3/3/25

[3] Sub-Challenge 3: Optic Disc and Fovea Detection. (2019). Retrieved from https://idrid.grand-challenge.org/Localisation/

[4] Salz, D. A., & Witkin, A. J. (2015). Imaging In Diabetic Retinopathy. Middle East African Journal of Ophthalmology,22(2),145-150.

[5] Medhi, J. P., & Dandapat, S. (2016). An effective fovea detection and automatic assessment of diabetic maculopathy in color fundus images. Computers in Biology and Medicine,74, 30-44. doi:10.1016/j.compbiomed.2016.04.007

[6] Winder, R., Morrow, P., McRitchie, I., Bailie, J., & Hart, P. (2009). Algorithms for digital image processing in diabetic retinopathy. Computerized Medical Imaging and Graphics,33(8), 608-622.

[7] Gnagwani, R., Lian, J., McGhee, S., Wong, D., & Li, K. (20165). Diabetic retinopathy screening: Global and local perspective. Hong Kong Medical Journal,22(5), 486-496.

Appendix A - Results

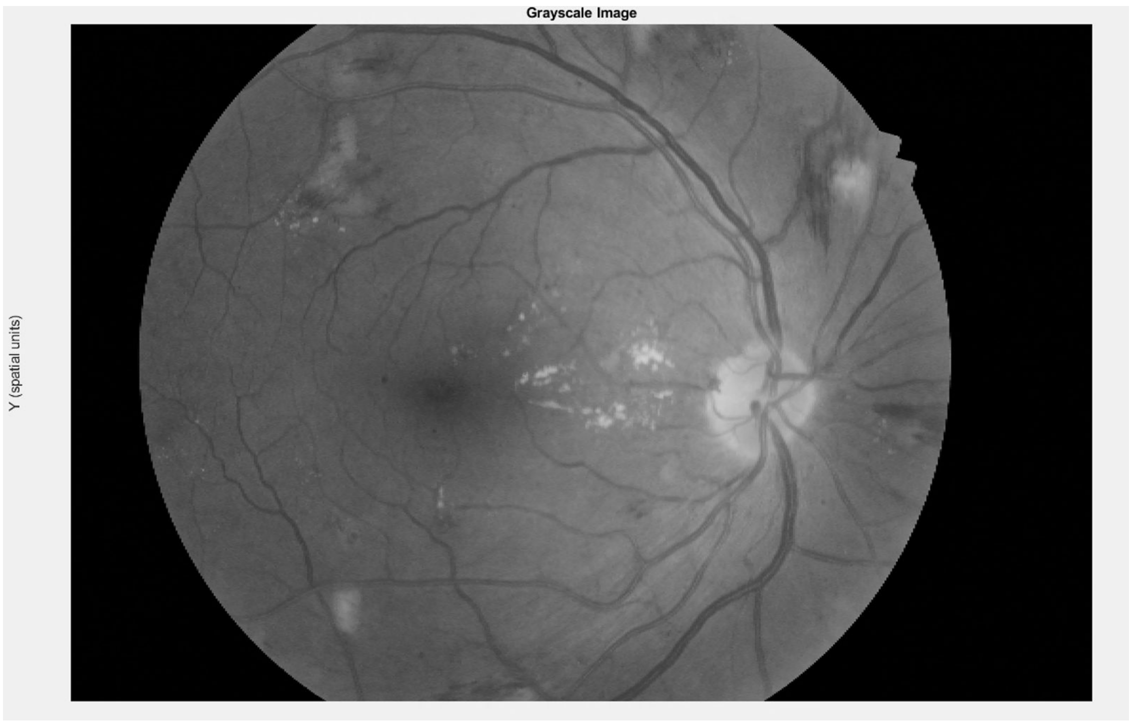

Figure 4. Grayscale Image

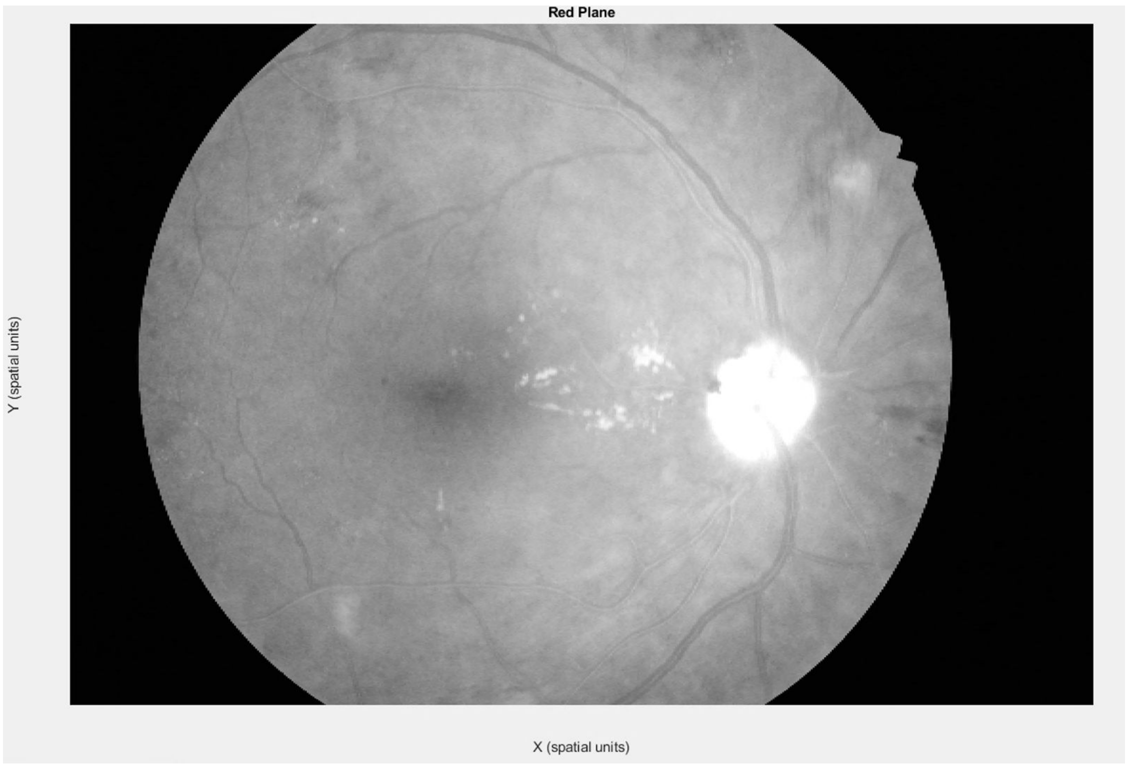

Figure 5. Red plane image

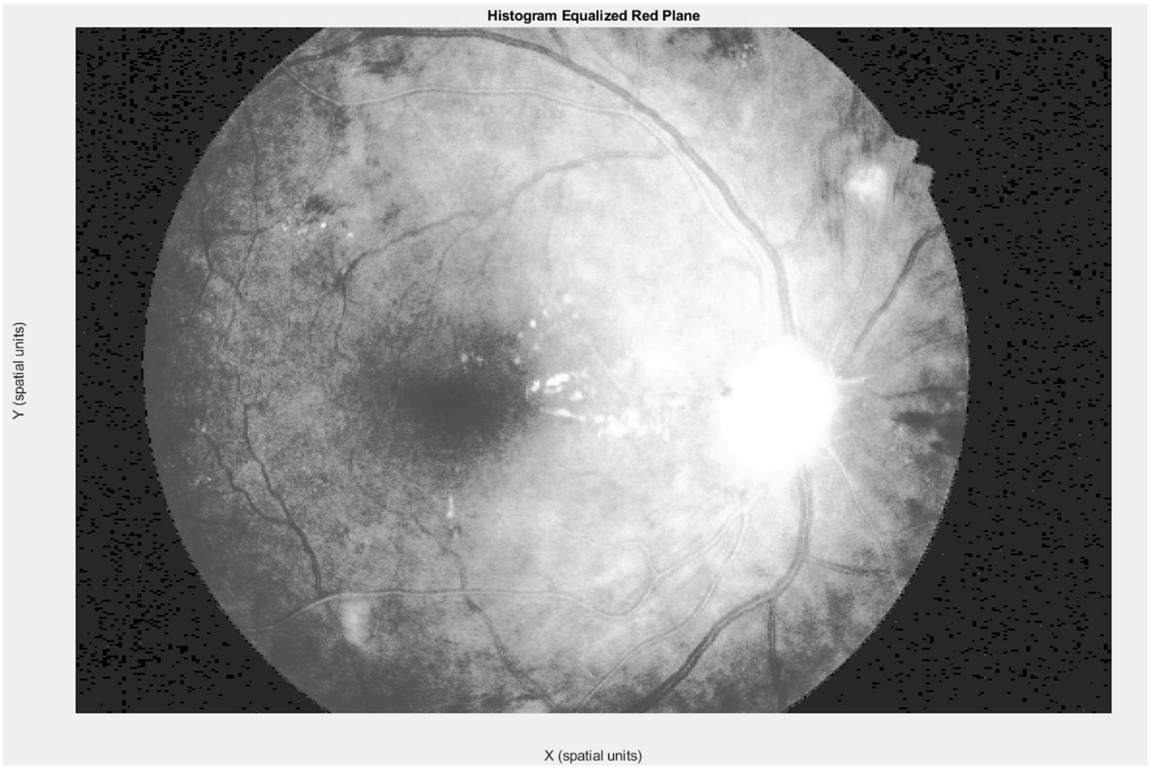

Figure 6. Histogram equalized red plane image

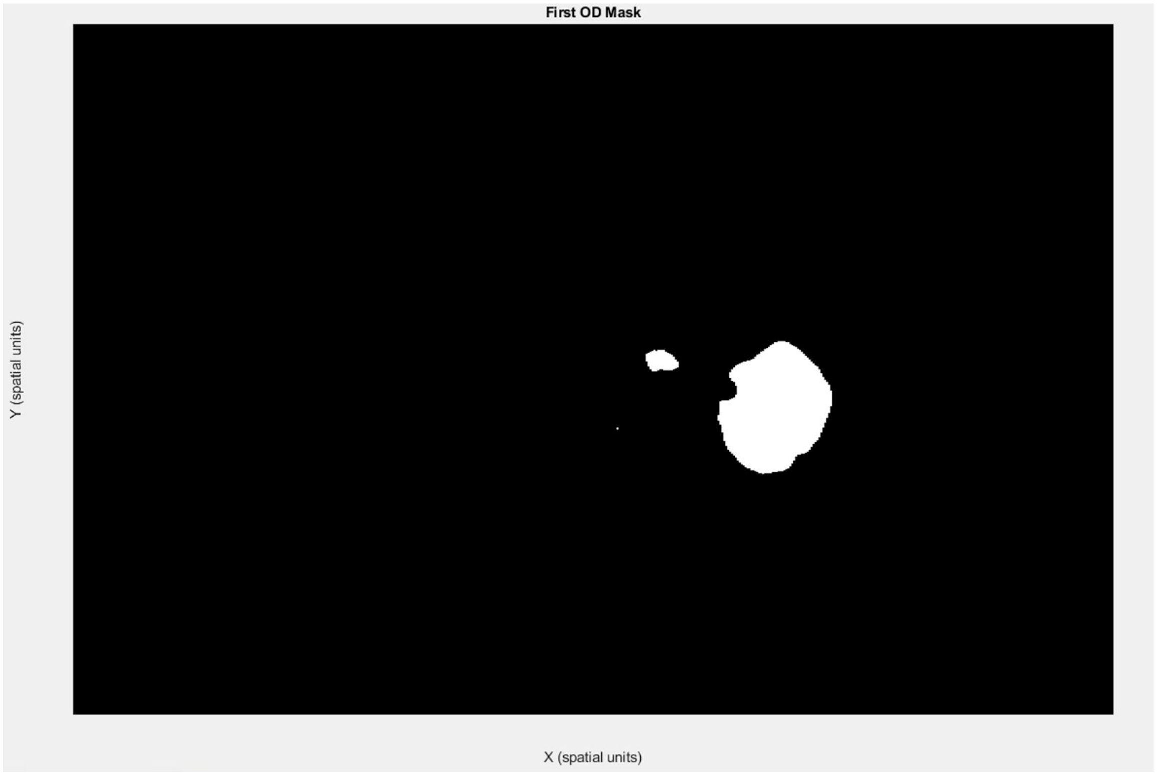

Figure 7. First attempt at OD mask

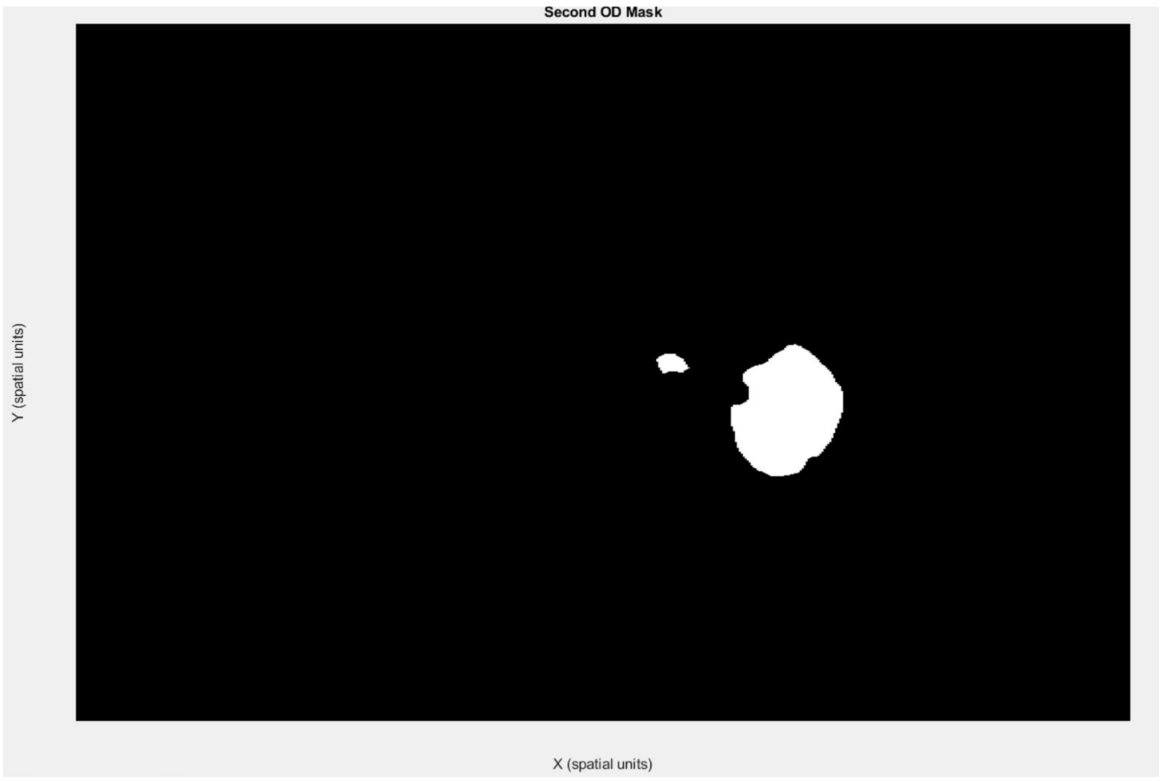

Figure 8. Second attempt at OD mask

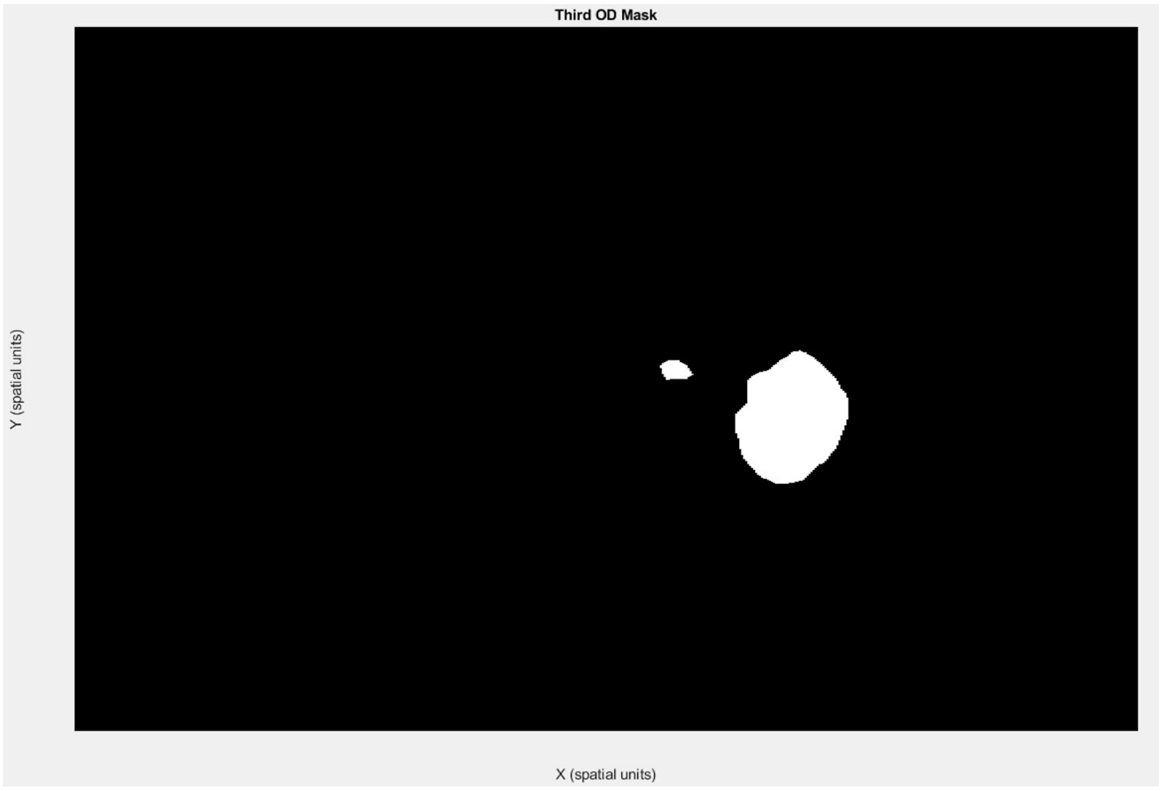

Figure 9. Third attempt at OD mask

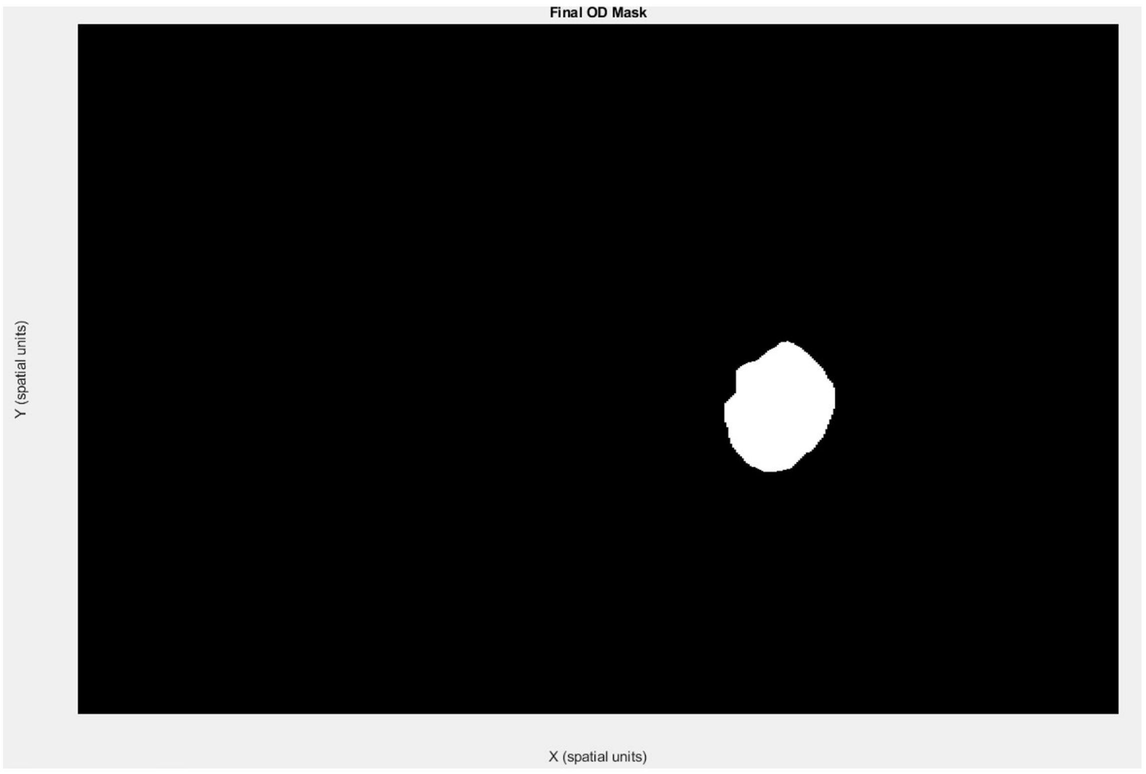

Figure 10. Final form of OD mask

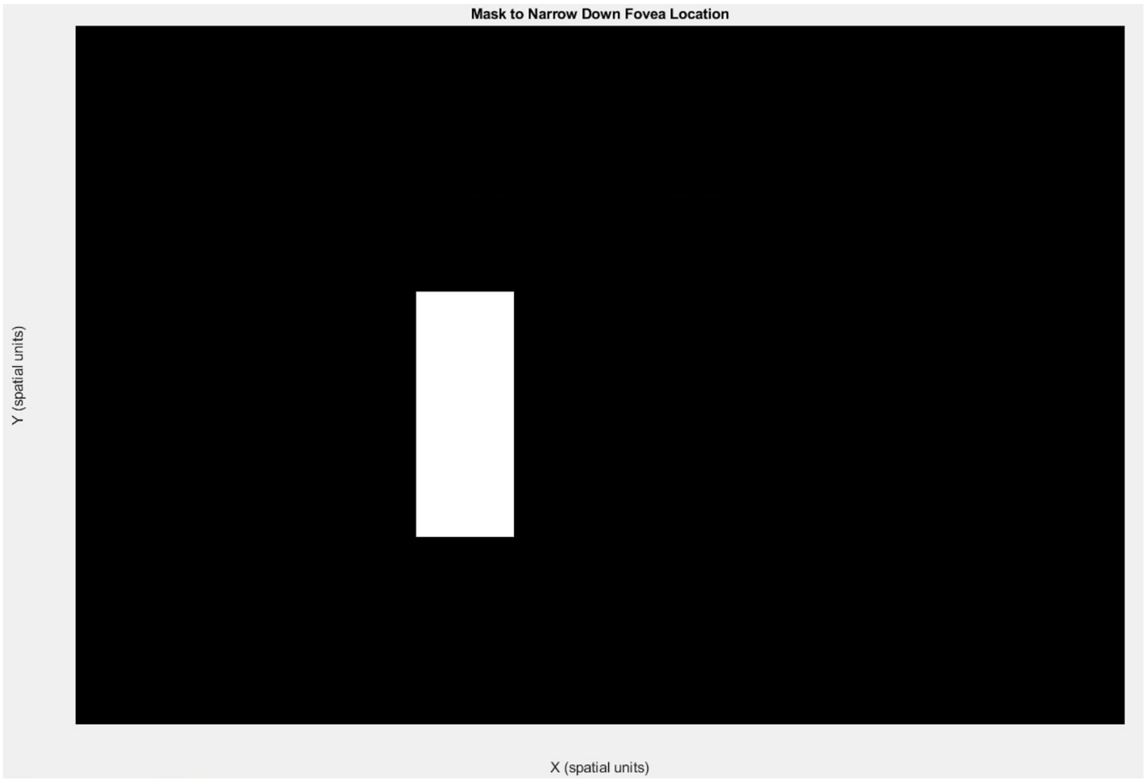

Figure 11. Region of interest mask for fovea

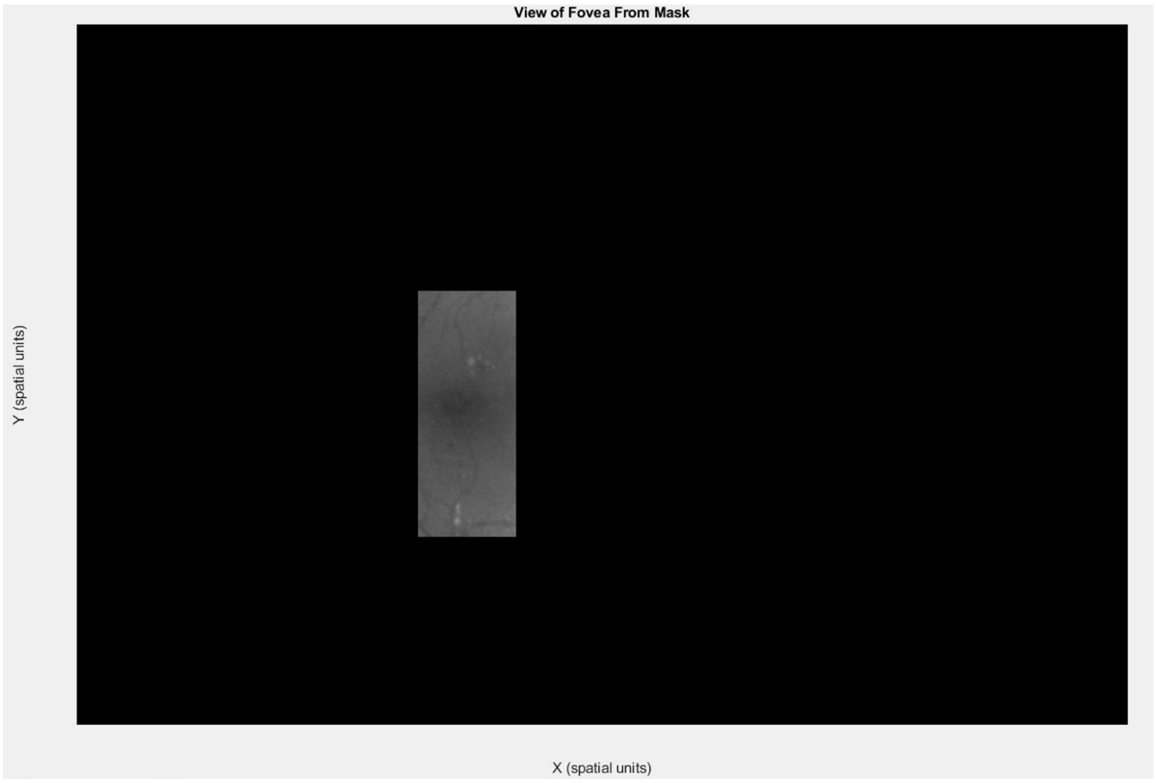

Figure 12. Masked grayscale image showing only region of interest for fovea

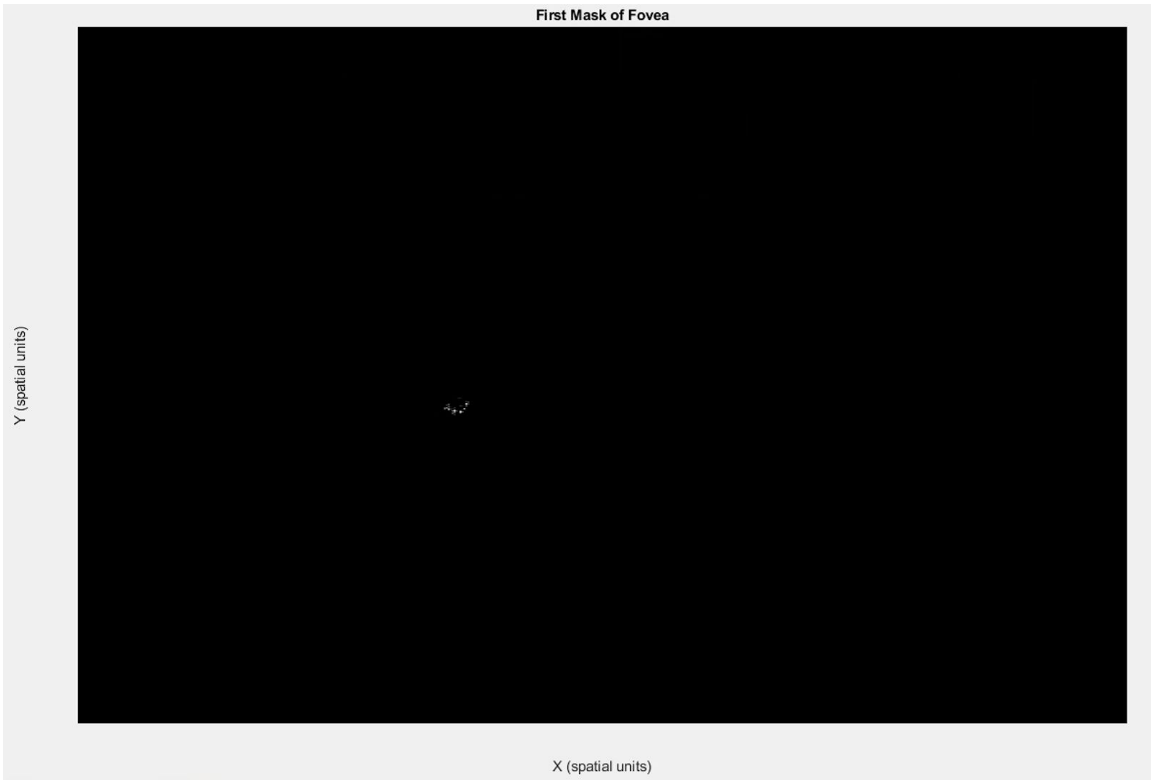

Figure 13. First attempt at fovea mask

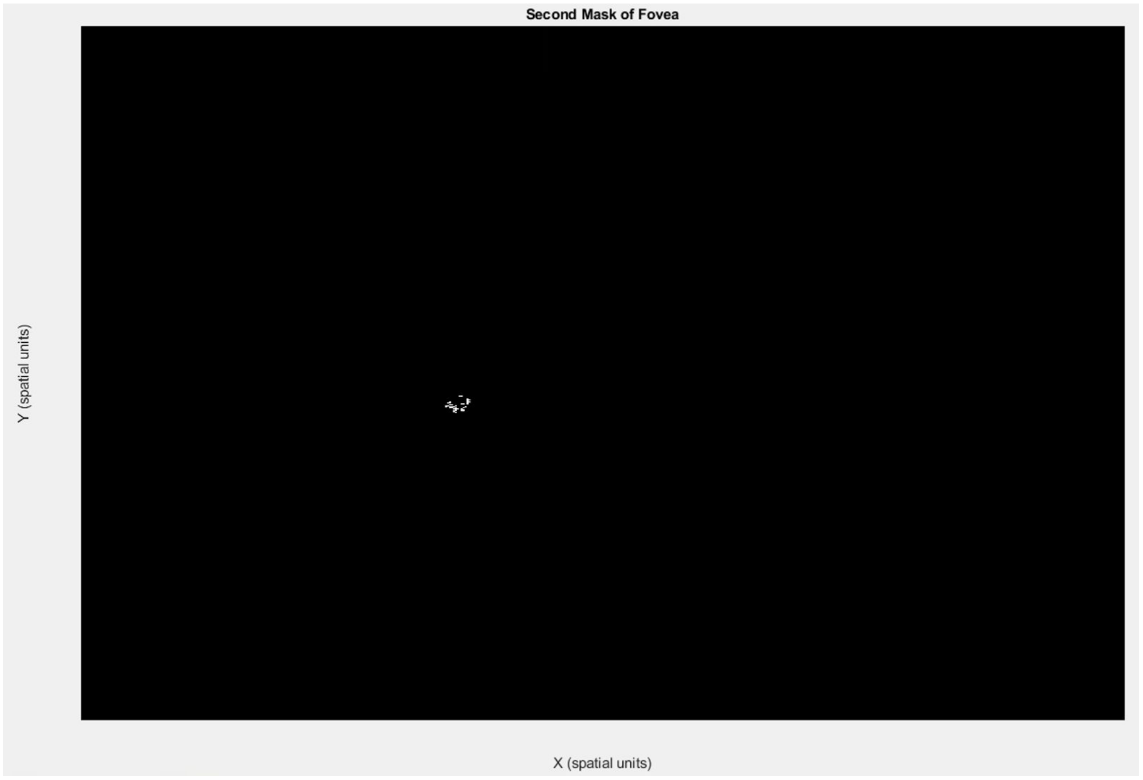

Figure 14. Second attempt at fovea mask

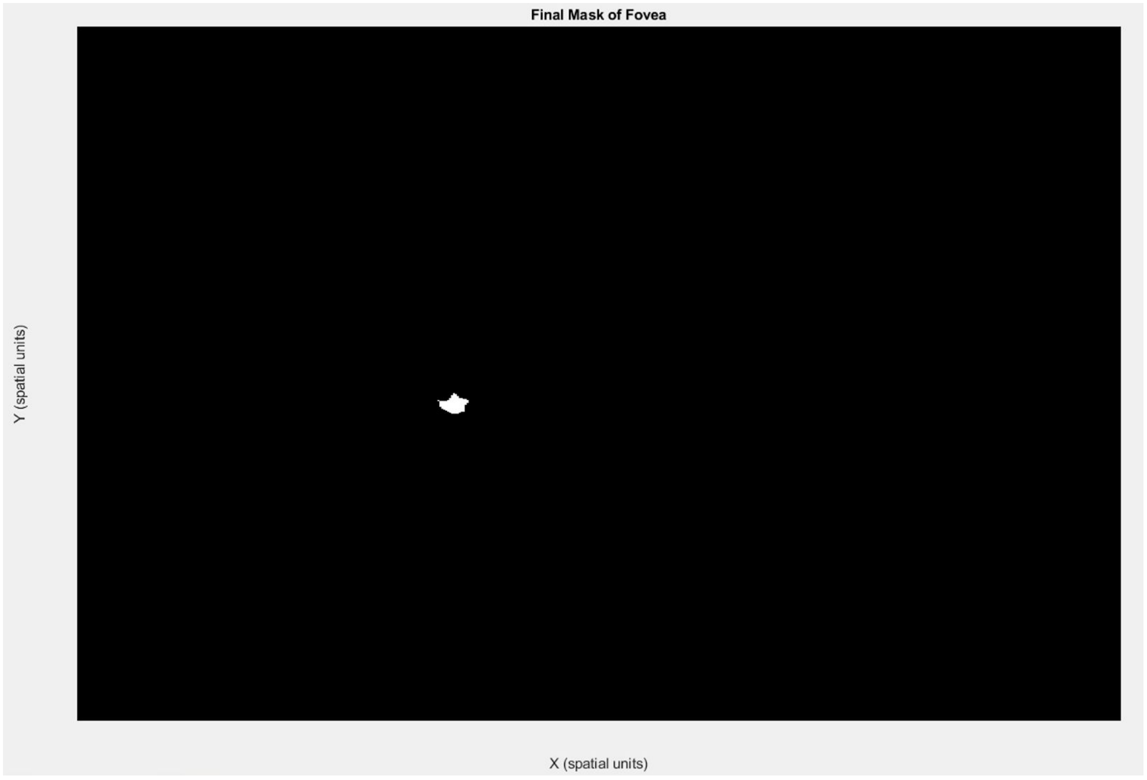

Figure 15. Final version of fovea mask

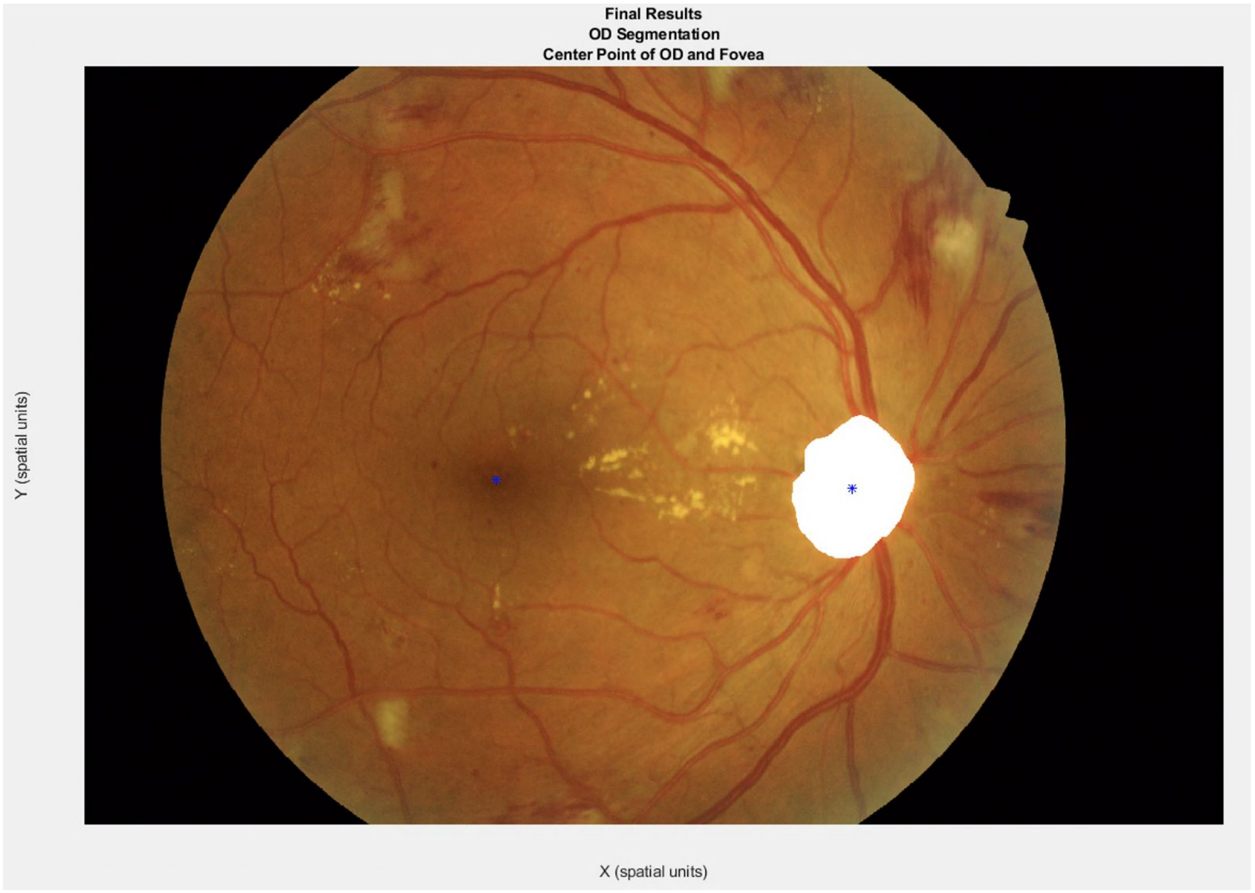

Figure 16. Final resultant image showing OD segmentation, OD localization, and fovea localization

Appendix - Source Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Title: Final Project: Fovea and Optic Disc Localization
% Filename: Tipton_EGR532_FinalProject.m
% Author: Natalie Tipton
% Date: 4/18/19
% Instructor: Dr. Rhodes
% Description: This algorithm solves the problem presented by IDRiD in their
%   challenge hosted on Grand Challenges in Image Processing. This will
%   take a folder with as many fundus scans of the retina as desired. It
%   will segment the optical disc and localize the optic disc and fovea by
%   finding the center points.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%clear all variables and command window
clear; clc;

%obtain directory path from user
path = uigetdir('*.tif*');
orig_path = cd(path);  %change directory so MATLAB can find files
files = dir('IDRiD*.jpg*');   %open all .tif or .tiff files in directory with IS in name
mask = ismember({files.name}, {'.', '..'});
files(mask) = [];   %get rid of . and .. directories
[num_files,z] = size(files);      %determine number of files read

%complete loop for each image read from directory
for n = 1:num_files
        %after each image, clear all variables except cumulative data variables
        if n > 1
        clearvars -except im n numfiles files path orig_path centroid_od centroid_f
        end
        %read in all images in directory
        cd(path);
        im{n} = imread(files(n).name);
        cd(orig_path);

        %use nth image in directory for current analysis
        image = im{1,n};
        image = im2double(image);

%obtain grayscale image
        image_gray = rgb2gray(image);
        [row, col] = size(image_gray);

        %obtain red plane of original image and perform histogram equalization
        red = image(:,:,1);
        red_eq = histeq(red);

        %blur equalized image, create a threshold, and binarize
        red_eq_blur = blur_gray(red_eq, 50);
        thresh_od = 0.99 * max(red_eq(:));
        bin_od = im2bw(red_eq_blur, thresh_od);

        %blur binarized image to reduce noise, create threshold, and binarize
        %again
        bin2_od = blur_gray(bin_od,10);
        thresh2_od = .9 * max(bin2_od(:));
        bin2_od = im2bw(bin2_od, thresh2_od);

        %fill in gaps in circular OD mask
        se = strel('disk',100);
        last_od = imclose(bin2_od,se);

        %remove all white regions aside from largest one
od_mask = bwareafilt(last_od,1);

        %Place OD mask on top of original image
        for x = 1:row
                for y = 1:col
                        for k = 1:3
                                if od_mask(x,y) == 1
                                        fin(x,y,k) = 1;
                                else
                                        fin(x,y,k) = image(x,y,k);
                                end
                        end
                end
```

```
end

%find centroid of OD
s = regionprops(od_mask, 'centroid');
centroid_od(n,:) = cat(1,s.Centroid);

%create mask based on location of OD to find fovea
%mask exists in range from 1200 to 1500 pixels to the side of the OD and
%500 pixels above and below the OD location in the X range from above
if centroid_od(n,1) < 2000      %if OD located on left side of image
        right_lim = centroid_od(n,1) + 1500;
        left_lim = centroid_od(n,1) + 1100;
        top_lim = centroid_od(n,2) + 500;
        bot_lim = centroid_od(n,2) - 500;
        for x = 1:row
                for y = 1:col
                        if y < right_lim && y > left_lim
                                if x < top_lim && x > bot_lim
                                        fovea_mask(x,y) = 1;
                                else
                                        fovea_mask(x,y) = 0;
                                end
                        else
                                fovea_mask(x,y) = 0;
                end
         end
end

else                    %if OD located on right side of image
        left_lim = centroid_od(n,1) - 1500;
        right_lim = centroid_od(n,1) - 1100;
        top_lim = centroid_od(n,2) + 500;
        bot_lim = centroid_od(n,2) - 500;
        for x = 1:row
                for y = 1:col
                        if y < right_lim && y > left_lim
                                if x < top_lim && x > bot_lim
                                        fovea_mask(x,y) = 1;
                                else
```

```
                                        fovea_mask(x,y) = 0;
                                end
                        else
                                fovea_mask(x,y) = 0;
                        end
                end
        end
end

%apply mask to grayscale image
fovea_loc = image_gray.* fovea_mask;

%create threshold and binarize image to create mask over fovea location
thresh = 1.1 * min(fovea_loc(fovea_loc~=0));
for x = 1:row
        for y = 1:col
                if fovea_loc(x,y) < thresh && fovea_loc(x,y) ~= 0
                        fovea_bin(x,y) = 1;
                 else
                        fovea_bin(x,y) = 0;
                end
        end
end

%blur original fovea mask, find another threshold and re-binarize
fovea_bin = blur_gray(fovea_bin, 3);
thresh_bin = 0.1 * max(fovea_bin(:));
fovea_bin_fin = im2bw(fovea_bin, thresh_bin);

%fill in gaps in circular region of fovea and remove all but largest
%region
se = strel('disk',25);
last_fov = imclose(fovea_bin_fin,se);
last_fov = bwareafilt(last_fov,1);

%apply mask where fovea is located to the grayscale image
fov_fin = image_gray .* last_fov;

%find centroid of fovea
```

```
s2 = regionprops(last_fov, 'centroid');
centroid_f(n,:) = cat(1,s2.Centroid);

%plot all steps of the algorithm in images with titles and axis labels
figure; imshow(image_gray);
title('Grayscale Image');
xlabel('X (spatial units)'); ylabel('Y (spatial units)');
figure; imshow(red);
title('Red Plane');
xlabel('X (spatial units)'); ylabel('Y (spatial units)');
figure; imshow(red_eq)
title('Histogram Equalized Red Plane');
xlabel('X (spatial units)'); ylabel('Y (spatial units)');
figure; imshow(bin_od)
title('First OD Mask');
xlabel('X (spatial units)'); ylabel('Y (spatial units)');
figure; imshow(bin2_od)
title('Second OD Mask');
xlabel('X (spatial units)'); ylabel('Y (spatial units)');
figure; imshow(last_od);
title('Third OD Mask');
xlabel('X (spatial units)'); ylabel('Y (spatial units)');
figure; imshow(od_mask);
title('Final OD Mask');
xlabel('X (spatial units)'); ylabel('Y (spatial units)');
figure; imshow(fovea_mask)
title('Mask to Narrow Down Fovea Location');
xlabel('X (spatial units)'); ylabel('Y (spatial units)');
figure; imshow(fovea_loc)
title('View of Fovea From Mask');
xlabel('X (spatial units)'); ylabel('Y (spatial units)');
figure; imshow(fovea_bin)
title('First Mask of Fovea');
xlabel('X (spatial units)'); ylabel('Y (spatial units)');
figure; imshow(fovea_bin_fin)
title('Second Mask of Fovea');
xlabel('X (spatial units)'); ylabel('Y (spatial units)');
figure; imshow(last_fov)
title('Final Mask of Fovea');
```

```
        xlabel('X (spatial units)'); ylabel('Y (spatial units)');
        figure; imshow(fin)
        hold on
        plot(centroid_od(:,1),centroid_od(:,2),'b*')
        plot(centroid_f(n,1),centroid_f(n,2),'b*')
        hold off
        title({'Final Results','OD Segmentation','Center Point of OD and Fovea'});
        xlabel('X (spatial units)'); ylabel('Y (spatial units)');

end

%round centroid (x,y) values
centroid_od = round(centroid_od);
centroid_f = round(centroid_f);

%export OD centroid data to a spreadsheet
T = table(centroid_od);
T(1:n,:);
filename = 'Project_Demo.xlsx';
writetable(T, filename, 'Sheet', 1, 'Range', 'A1');

%export fovea centroid data to same spreadsheet
T2 = table(centroid_f);
T2(1:n,:);
filename2 = 'Project_Demo.xlsx';
writetable(T2, filename2, 'Sheet', 1, 'Range', 'D1');

function im_blur = blur(im, neighbor)

%This function blurs a given image using a neighborhood averaging filter
%with neighborhood size determined by input variable

 im_blur = conv2(im, ones(neighbor,neighbor)/(neighbor^2), 'same');
```