**Autonomous Snake Algorithm for Eye Images**

## Algorithm Logic

|  |  |
| --- | --- |
| 1. **Steps to remove the eyelid (myActiveConour.m)** | **Output** |
| Upload the raw RGB image. |  |
| Convert the eye to grayscale using **rgb2gray()**[1] to place mask over the image. |  |
| Place mask using **zeros()**over the gray eye and Use **activecontour()**[2] to produce a black and white (BW) output of the eye’s surroundings eliminated. |  |
| Compare the BW image in the previous step with the original image. If a pixel is black in the BW image, then change the value of that pixel in the original RGB image to white. |  |

|  |  |
| --- | --- |
| 1. **Steps to find the pupil’s centre (centerFinder.m)** | **Output** |
| Take the output of **(1)** as an input. Increase the image’s intensity using **imadjust()**[3]. Increase the image’s sharpness and the radius of the boundary lines using **imsharpen()**[4]. |  |
| Convert the image to BW using **im2bw()**. Smooth the BW image using 2D convolution **conv2()**[5]. |  |
| Detect circles of radius between 100 and 150 using **imfindcircles()**[6].If multiple circles have been detected, return the centre coordinates and radius of the larger one. |  |

|  |  |
| --- | --- |
| 1. **Steps to remove the pupil (removePupil.m)** | **Output** |
| Take the output of **(1)** as an input. Increase the image’s intensity using **imadjust()**. Increase the image’s sharpness and the radius of the boundary lines using **imsharpen()**. |  |
| Convert the image to BW using **im2bw()**. Clean the BW image by removing tiny dots using **bwareaopen()**[7]. |  |
| Compare the BW image in the previous step with the original image. If a pixel is white in the BW image, then change the value of that pixel in the original RGB image to white. |  |
| The orange part from the result of the previous step needs to be removed. Using the function **centerFinder.m** presented in **(2)**, take the image from the previous step as an input. Detect the pupil circle and remove 25 from its radius (using trail and error). |  |
| Compute the Euclidean distance of each point. If distance<radius-25, change that pixel to white. |  |
| 1. **Steps to clean the image (myCleaner.m)** | **Output** |
| Take the output of **(3)** as an input. Convert the image to BW and use **bwareaopen()** to clean tiny dots and segments on the image. |  |
| Compare the BW image in the previous step with the original image. If a pixel is white in the BW image, then change the value of that pixel in the original RGB image to white. Return the final result. |  |

## Software

|  |
| --- |
| 1. **snake.m (this is the main)** |
| originalEye = imread('P6a.png'); % P2a,P3a,P4a,P5a,P6a  figure()  imshow(originalEye)  title('Orginal Eye');    contouredEye = myActiveContour(originalEye,1000);    test = removePupil(contouredEye);    cleanedImg = myCleaner(test,10000); |

|  |
| --- |
| 1. **myActiveContour.m** |
| function contouredImg = myActiveContour(inputImg,Iterations)  %keep Iterations=1000    grayEye=rgb2gray(inputImg);  % figure()  % imshow(grayEye)  % title('Gray Eye');    % Active contour  mask = zeros(size(grayEye));  mask(25:end-25,25:end-25) = 1;  % imshow(mask)  % title('Initial Contour Location')    bw = activecontour(grayEye,mask,Iterations);    % figure()  % imshow(bw)  % title('Segmented Image, ?? Iterations')    %Extractiing exterior eye  [r c] = size(bw);    for i=1:r  for j=1:c  if bw(i,j)== 0  inputImg(i,j,1)=255;  inputImg(i,j,2)=255;  inputImg(i,j,3)=255;  end  end  end    % figure()  % imshow(inputImg)  % title('Contoured and Cropped Image');  contouredImg=inputImg;  end |

|  |
| --- |
| 1. **centerFinder.m** |
| function noPupilImg = centerFinder(inputImg)    % figure()  % imshow(inputImg)  % title('Orginal Eye');    low = 0.4;  high = 0.9;  K = imadjust(inputImg,[low high],[]); % I is double  b = imsharpen(K,'Radius',15,'Amount',10);    % figure()  % imshow(b)  % title('b ');    bw=im2bw(b,0.05);    % figure()  % imshow(bw)  % title('bw Eye');    windowSize = 51;  kernel = ones(windowSize) / windowSize ^ 2;  blurryImage = conv2(single(bw), kernel, 'same');  bw = blurryImage > 0.5; % Rethreshold    % figure()  % imshow(bw)  % title('cir Eye');    % d=imdistline;  [centers, radii] = imfindcircles(bw,[100 150],'ObjectPolarity','dark','Sensitivity',0.92);  numCircles = length(centers);  % viscircles(centers, radii,'EdgeColor','b');    [MaxRadii,MaxRadiiIndex]=max(radii);  bigCircleCenter=centers(MaxRadiiIndex,:);    x=bigCircleCenter(2);  y=bigCircleCenter(1);  MaxRadii=MaxRadii-25;    noPupilImg=[x,y,MaxRadii];  end |

|  |
| --- |
| 1. **removePupil.m** |
| function test = removePupil(inputImg)    inputImageCopy=inputImg;  inputImageCopy2=inputImg;  inputImageCopy3=inputImg;    % figure()  % imshow(inputImg)  % title('Orginal Eye IN REMOVEPUPIL');    low = 0.4;  high = 0.9;  K = imadjust(inputImg,[low high],[]); % I is double  b = imsharpen(K,'Radius',15,'Amount',10);    bb = imsharpen(K,'Radius',1,'Amount',2);    % figure()  % imshow(b)  % title('b ');    bw=im2bw(b,0.05);  bw2=im2bw(bb,0.05);    bw2 = bwareaopen(imcomplement(bw2), 10000);  bw2=imcomplement(bw2);    % figure()  % imshow(bw);  % title('bw pupil');    % figure()  % imshow(bw2);  % title('bw2 pupil');    [r c] = size(bw);    for i=1:r  for j=1:c  if bw2(i,j)== 1  inputImageCopy(i,j,1)=255;  inputImageCopy(i,j,2)=255;  inputImageCopy(i,j,3)=255;  end  end  end    for i=1:r  for j=1:c  if inputImageCopy(i,j)~= 255  inputImageCopy2(i,j,1)=255;  inputImageCopy2(i,j,2)=255;  inputImageCopy2(i,j,3)=255;  end  end  end    % figure()  % imshow(inputImageCopy2);  % title('inputImageCopy2');    cc = centerFinder(inputImageCopy3);    for i=1:r  for j=1:c  if sqrt((cc(1)-i)^2+(cc(2)-j)^2)<=cc(3)  inputImageCopy2(i,j,1)=255;  inputImageCopy2(i,j,2)=255;  inputImageCopy2(i,j,3)=255;  end  end  end    % figure()  % imshow(inputImageCopy2);  % title('end');    test=inputImageCopy2;  end |

|  |
| --- |
| 1. **myCleaner.m** |
| function cleanedImg = myCleaner(inputImg,rate)  %keep rate=10000    bw=im2bw(inputImg,0.999999999999);    % figure()  % imshow(bw)  % title('bw');    bwClean = bwareaopen(imcomplement(bw), rate);  bwClean=imcomplement(bwClean);    % figure()  % imshow(bwClean)  % title('bwClean');    [r c] = size(bw);    for i=1:r  for j=1:c  if bwClean(i,j)==1  inputImg(i,j,1)=255;  inputImg(i,j,2)=255;  inputImg(i,j,3)=255;  end  end  end    % figure()  % imshow(bwClean)  % title('bwClean');    % figure()  % imshow(inputImg)  % title('inputImg');    cleanedImg=inputImg;  end |

# Implementation and Testing

|  |  |
| --- | --- |
| **P6a.png** | |
| **Before:** | **After:** |
| A picture containing indoor, silver  Description automatically generated | A close-up of a human eye  Description automatically generated with low confidence |

|  |  |
| --- | --- |
| **P5a.png** | |
| **Before:** | **After:** |
| A close-up of an eyeball  Description automatically generated with low confidence | A close-up of a snail  Description automatically generated with low confidence |

|  |  |
| --- | --- |
| **P4a.png** | |
| **Before:** | **After:** |
| A picture containing indoor  Description automatically generated | A close up of a garlic  Description automatically generated with low confidence |

|  |  |
| --- | --- |
| **P3a.png** | |
| **Before:** | **After:** |
| A close up of a person's eye  Description automatically generated with medium confidence | A close-up of a snail  Description automatically generated with medium confidence |

|  |  |
| --- | --- |
| **P2a.png** | |
| **Before:** | **After:** |
|  |  |

|  |  |
| --- | --- |
| **Image name** | **Time to process** |
| P6a.png | Elapsed time is 12.132924 seconds. |
| P5a.png | Elapsed time is 9.245235 seconds. |
| P4a.png | Elapsed time is 9.698004 seconds. |
| P3a.png | Elapsed time is 9.839956 seconds. |
| P2a.png | Elapsed time is 12.559268 seconds. |
| **Average processing time:** | **10.695077** |

|  |
| --- |
| **Results’ discussion:** |
| * The nature of images is arguably difficult to process as most of the borders in the image blend. Therefore, one major challenge in tackling this problem is coming up with a technique to make the initial image have borders. * Output for each image slightly varied. Some outputs had unwanted areas considered in the final image while some outputs had wanted areas removed. * Overall, the algorithm can generalize well considering that the method used to produce the output is naïve and does not make use of any techniques such as machine learning. |

# References

1. [online] Available at: <<https://www.mathworks.com/help/matlab/ref/rgb2gray.html>> [Accessed 22 May 2022].
2. [online] Available at: <<https://www.mathworks.com/help/images/ref/activecontour.html>> [Accessed 22 May 2022].
3. [online] Available at: <<https://www.mathworks.com/help/images/ref/imadjust.html>> [Accessed 22 May 2022].
4. [online] Available at: <<https://www.mathworks.com/help/images/ref/imsharpen.html>> [Accessed 22 May 2022].
5. [online] Available at: <<https://www.mathworks.com/help/matlab/ref/conv2.html>> [Accessed 22 May 2022].
6. [online] Available at: <<https://www.mathworks.com/help/images/ref/imfindcircles.html>> [Accessed 22 May 2022].
7. [online] Available at: <<https://www.mathworks.com/help/images/ref/bwareaopen.html>> [Accessed 22 May 2022].