MV_Exam_TaherVora

Problem 1. An image called “dyes.png” is given to you. In this image, two dyes are diffusing through the system, one from the top (green) and one from the bottom (red). Your job in this problem is to find the boundary of diffusions in this picture.

A. How would you find the boundary of the diffusion of each dye? (Note: If you have multiple answers in mind, break them apart and explain each one separately.) Explain each solution/algorithm in detail.

B. Implement your answer using any programming language you want and save the resulting image.

Solution : I have tried various approaches.

->First I tried to convert image into three different planes (RGB) and applied thresholding to extract red and green plane from it but the red color was too scattered so didn’t found any solution from this approach.

->The Second approach I thot was: playing with color pixels. I thought I could find a region(boundary) where intensity of green and red color would be less and black would be higher so that; that differences in intensity would give me a boundary. But wasn’t able to implement that too.

->The final approach which gave me a boundary was:

- Read the image.
- Apply thresholding.
- Applied **bwareaopen** to remove all the noise.(Top layer and bottom layer had high density(formed cluster) and middle layer had lower density so it was counted as noise and center portion was removed)
- Applied **bwboundaries** to get the boundaries of both the layers.

Code:

```
image = imread('dyes.png');
figure,imshow(image);

level = 0.04;
bin = im2bw(image, level);
figure, imshow(bin);
im3=bwareaopen(bin,200);
figure,imshow(im3);
[T,S] = bwboundaries(im3,'noholes');
figure,imshow(S)
hold on
for k = 1:length(T)
  boundary = T{k};
  plot(boundary(:,2),boundary(:,1),'r','LineWidth',1)
end
```

Output:

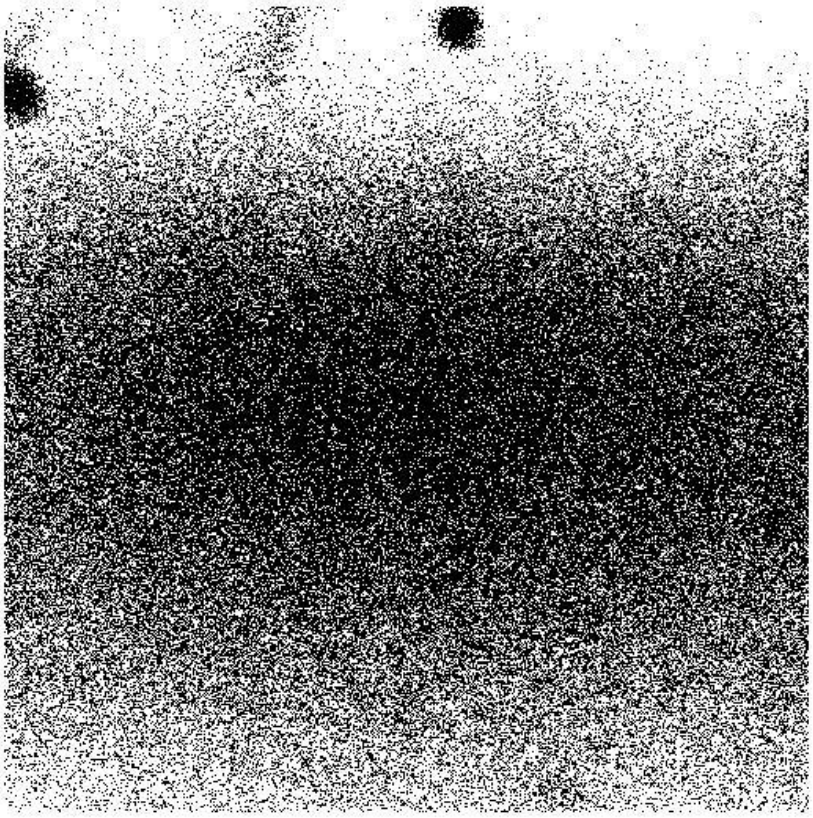

Figure 1 : Applied Thresholding

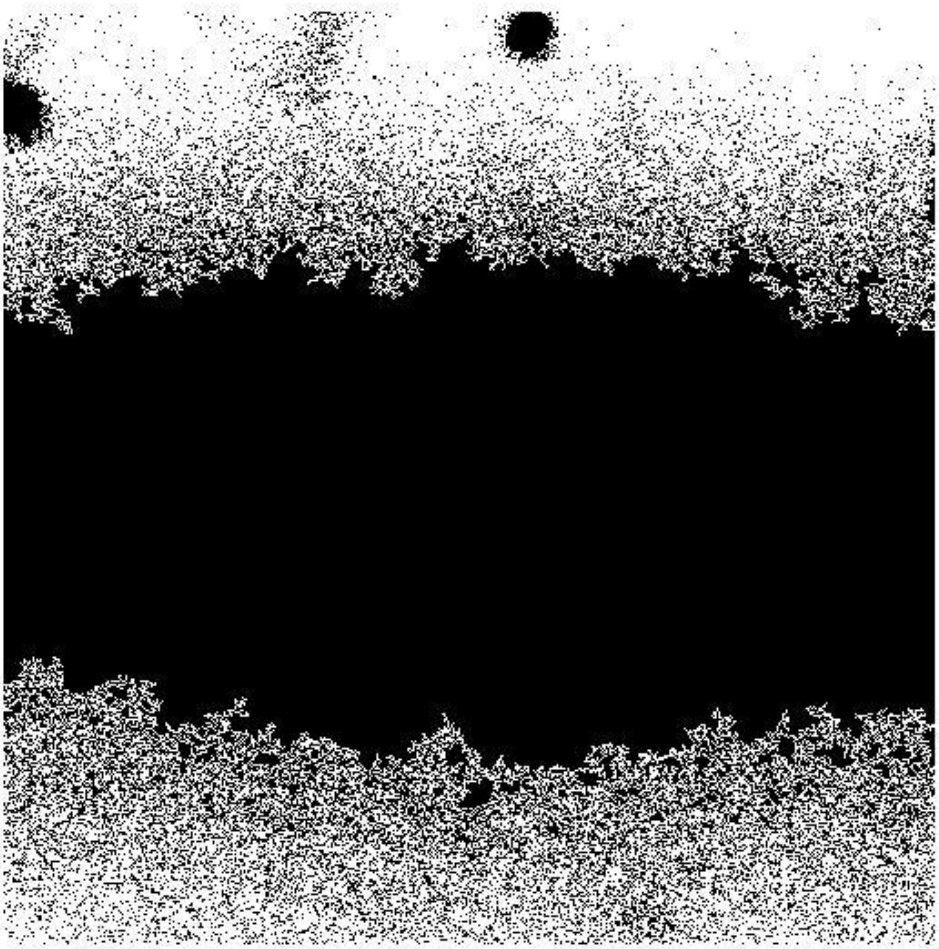

Figure 2 : Applied Bwareaopen (remove noise)

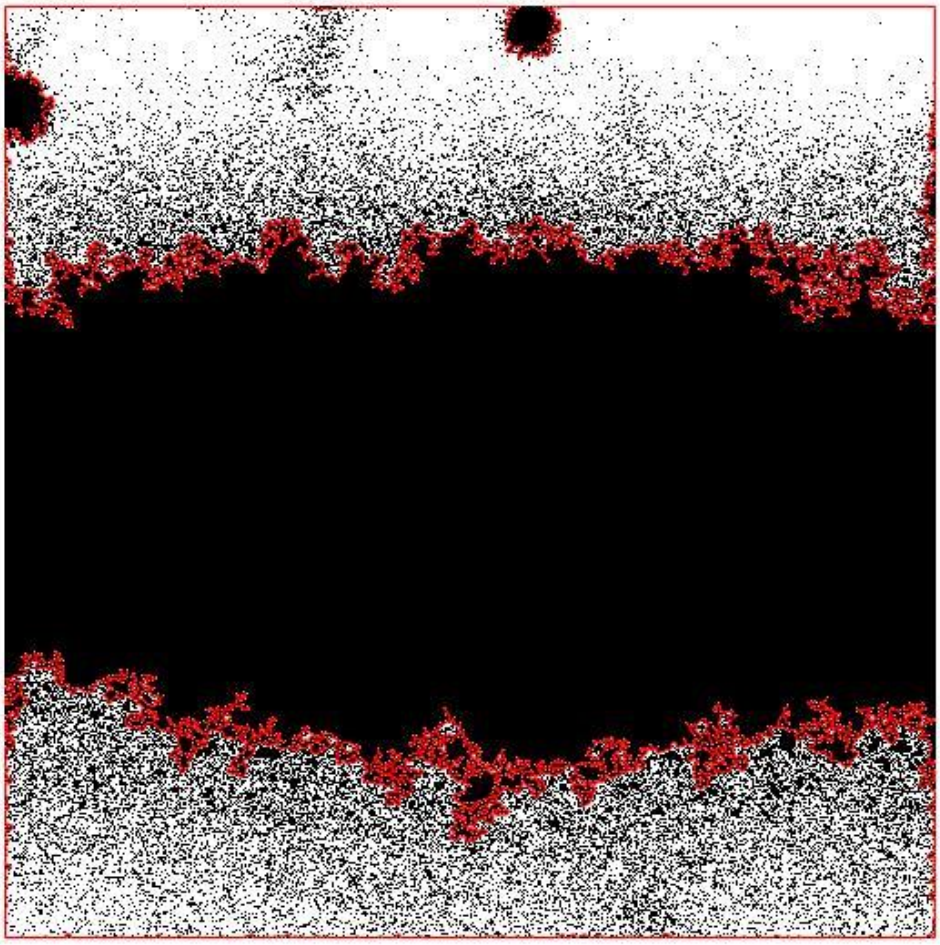

Figure 3: Applied bwboundaries

In final image:

- The upper layer is GREEN.
- Lower layer is RED.

Problem 2. A set of 2500 tiny grayscale images (28x28 pixels each) that represent handwritten digits 0 to 4 and their labels are given to you. The images are stored in "digits.mat" as a matrix of size 784x2500 where each column represents an image. The labels of the images are saved in "labels.mat". Design a model that could recognize the digit value of any given image.

A. How would you create your model to classify the digits? (Note: If you have multiple answers in mind, break them apart and explain each one separately.) Explain each solution/algorithm in detail.

B. Implement your answer using any programming language you want and save the resulting accuracy rate.

Solution:

The different approaches which can be used are:

- CNN
- KNN
- Gaussian etc.

I had some basic idea about CNN(studied it because I am using for my project work).

There are 3 layers:

- Convolution: Applies convolutional filters.
- ReLU: Converts negative values to 0 and maintains positive values.
- Pooling: Simplifies the output by performing nonlinear downsampling and reducing the number of parameters that the network needs to learn.

Implemented KNN:

Code:

Took 500 digit dataset.

```
%% Took help from Lab4.
% Extracted images from matrix.
for i=1:500
    im = data(:,i);
    im = reshape(im,28,28);
    All(:,:,i) = im;
end
All = reshape(All,size(All,1)*size(All,2),500);


%% k-partition and finding accuracy
k = 10;

c = cvpartition(500,'KFold',k);
for i=1:k;
    idxTrain = training(c,i);
    idxTest = test(c,i);
    TestIDs = find(idxTest == 1);
    TrainIDs = find(idxTrain == 1);
```

```
    nr_test = sum(idxTest);
    nr_train = sum(idxTrain);
    test_result = false(nr_test,1);
    for i = 1:nr_test
        testId = TestIDs(i);
        testIm = All(:,testId);
        dst = abs(All(:,idxTrain)-repmat(testIm,1,nr_train));
        dst = sum(dst);
        [~,mnIdx] = mink(dst,k);
        vote(i) = mode(labels(TrainIDs(mnIdx)));
        test_result(i) = vote(i) == labels(testId);
    end
end
disp(mean(test_result))
```

For different values of k(nearest neighbors) found different accuracies:

```
>> ExamQ2
    0.9000

    0.8800

    0.9000

    0.8600

    0.9600
```

For k=3,5,7,9,10 accuracies are as follows.