EE 569 HOMEWORK 4

Issued: 11/1/2015 Due: 11/29/2015

Name: Meiyi Yang

Email: meiyiyan@usc.edu

USC ID: 6761-0405-85

Problem 1. Optical Character Recognition (OCR)

1.1 Motivation

Optical Character Recognition (OCR) is used to analysis the shape in the images like typed, handwritten or printed text. In this problem, we use OCR to detect and analysis the character in four images by developing a decision tree. [1]

1.2 Algorithm and Implementation

1.2.1 OCR

To analysis and classify the character shape in the image, OCR technique is widely used to recognize the character. It includes several steps:

Preprocess the training and testing image: include binarizing the image and some other methods to get a more clearly character in the binarized image.

Segment the training and testing images.

Extract the feature of the training image.

Develop a decision trees by training features.

Use decision trees to determine the testing images.

Optional: Here I use the result of testing image to adjust the decision tree to get a more robust result.

1.2.2 Preprocessing

The methods of preprocessing used in problems are listed below. The details of each testing image used for preprocessing are listed in 1.4.1

Histogram equalization.

Image Denosing.

Morphological processing.

1.2.3 Segmentation

The segmentation method is implemented by counting 8-connectivity pathway and labeling referred from HW3, problem3. By labeling the different connected pathways. I get the total number of pathway of the image, and then segment the original image to the same sub-image if the pixels have the same label.

To get more useful result, I use two thresholds here to avoid segment too large or too small sub-image. Different images may have different thresholds.

Finally, the sub-image will be resized to the same size for later used.

1.2.4 Feature extraction

The types of feature extracted in the problem 1 are list below. The feature result and description is shown in 1.4.2.

Area.

Perimeter.

Circularity.

Symmetry. Symmetry includes two directions, x and y axis.

Euler Number.

Spatial Moment. [2]

1.2.5 Decision Tree

Decision tree is a tree-like model for supporting decision. It has two kinds of nodes. One is a class label, another is question of features [3]. The details of decision tree used in this problem is shown in 1.4.3.

1.3 Results

1.3.1 Decision Tree for reference Data

Label	Area	Peri	Circle	SymmetrX	SymmetrY	MomentX	MomentY	Euler	Result
E	259	584	0.00954298	0.656663	0.865146	0.511988	0.410521	1	E
S	239	636	0.00742495	0.692429	0.686587	0.511318	0.544289	1	S
E	263	582	0.00975707	0.663016	0.937219	0.507281	0.414734	1	E
D	254	632	0.00799115	0.671935	0.974557	0.508937	0.470984	0	D
P	235	572	0.0090258	0.707173	0.596203	0.380532	0.431255	0	P
L	148	316	0.018625	0.436775	0.645641	0.715338	0.353243	1	L
I	69	260	0.0128266	0.954959	0.978995	0.70413	0.639928	1	I
M	253	680	0.00687563	0.942037	0.695909	0.556818	0.504565	1	M
I	85	326	0.0100506	0.994255	0.994255	0.485235	0.533941	1	I
T	153	338	0.0168294	0.932489	0.632911	0.306307	0.508072	1	T
0	229	624	0.00739054	0.991232	0.992902	0.508624	0.512118	0	0
7	139	370	0.0127591	0.677593	0.311953	0.327518	0.541835	1	7

Total 12, 12, 100 %

Figure 1. Feature extraction for training image

The result of feature extraction is shown in Figure 1. The result of decision tree is shown in Figure2.

```
// Decision Tree classifier
string Classifier(double feature_euler, double feature_symmetry_X, double feature_symmetry_Y,
 double feature_moment_X, double feature_moment_Y, double feature_area, double
 feature_perimeter) {
    string res = "";
    if (feature_area < 100) {
        return "I";
    }
    if (feature_euler > 0) {
        if (feature_symmetry_X > 0.87) {
            if (feature_moment_X > 0.5)
                return "M";
            else
                return "T";
        } else if (feature_symmetry_X > 0.47 && feature_symmetry_X <= 0.87) {
            if (feature_symmetry_Y > 0.80) {
                    return "E";
            } else if (feature_symmetry_Y > 0.6 && feature_symmetry_Y <= 0.9) {
                if (feature_moment_Y > 0.55)
                    return "3";
                else if (feature_moment_X > 0.52)
                    return "2";
                else
                    return "5";
            } else if (feature_symmetry_Y > 0.35 && feature_symmetry_Y <= 0.6) {
                if (feature_moment_X > 0.5)
                    return "2";
                else
                    return "5";
            } else
                return "7";
        } else {
            if (feature_moment_X > 0.5)
                return "L";
            else
                return "1";
        }
    } else if (feature_euler == 0) {
        if (feature_symmetry_X > 0.89) {
                return "0";
        } else {
            if (feature_symmetry_Y > 0.85) {
                return "D";
            } else {
                if (feature_moment_X > 0.44) {
                    if (feature_moment_Y > 0.5) {
                        if (feature_perimeter > 630)
                            return "9";
                        else
                            return "4";
                    } else
                        return "6";
                } else
                    return "P";
            }
        }
    } else
        return "8";
}
```

Figure 2. Final decision tree

1.3.2 OCR Testing: Simple Cases

The final recognizing result of Test_ideal1 and Test_idea2 in shown in Figure 3 and 4.

Label	Area	Peri	Circle	SymmetrX	SymmetrY	MomentX	MomentY	Euler	Result
E	259	584	0.00954298	0.656663	0.865146	0.511988	0.410521	1	E
S	239	636	0.00742495	0.692429	0.686587	0.511318	0.544289	1	S
E	263	582	0.00975707	0.663016	0.937219	0.507281	0.414734	1	E
D	254	632	0.00799115	0.671935	0.974557	0.508937	0.470984	0	D
P	235	572	0.0090258	0.707173	0.596203	0.380532	0.431255	0	P
L	148	316	0.018625	0.436775	0.645641	0.715338	0.353243	1	L
I	69	260	0.0128266	0.954959	0.978995	0.70413	0.639928	1	I
M	253	680	0.00687563	0.942037	0.695909	0.556818	0.504565	1	M
I	85	326	0.0100506	0.994255	0.994255	0.485235	0.533941	1	I
T	153	338	0.0168294	0.932489	0.632911	0.306307	0.508072	1	T
0	229	624	0.00739054	0.991232	0.992902	0.508624	0.512118	0	0
7	139	370	0.0127591	0.677593	0.311953	0.327518	0.541835	1	7

Total 12, 12, 100 %

Figure 3. OCR for Test_idea1. Total character: 12. Correct rate: 100%

Label	Area	Peri	Circle	SymmetrX	SymmetrY	MomentX	MomentY	Euler	Result
S	233	634	0.00728429	0.786844	0.78757	0.511996	0.511137	1	S
P	211	530	0.00943932	0.682286	0.598857	0.386327	0.438839	0	P
E	270	618	0.00888376	0.711377	0.934474	0.517222	0.459556	1	E
E	272	616	0.00900777	0.715017	0.936519	0.510772	0.462757	1	E
D	245	628	0.00780651	0.788593	0.963115	0.515857	0.481694	0	D
L	142	320	0.017426	0.408716	0.653734	0.712746	0.364296	1	L
I	72	238	0.0159731	0.971738	0.971101	0.369444	0.694722	1	I
M	235	626	0.00753579	0.968238	0.695627	0.568234	0.522617	1	M
I	1	4	0.785398	0.972938	0.977626	0.515	0.515	1	I
T	145	330	0.0167321	0.981926	0.643647	0.317276	0.517621	1	T
3	223	612	0.0074819	0.784282	0.798461	0.506928	0.580336	1	3
1	118	298	0.0166978	0.304716	0.774614	0.400508	0.599407	1	1

Total 12, 12, 100 %

Figure 4. OCR for Test_idea1. Total character: 12. Correct rate: 100%

1.3.3 OCR Testing: Advanced Cases

The final recognizing result of Test_night and Test_shade in shown in Figure 5 and 6.

Label	Area	Peri	Circle	SymmetrX	SymmetrY	MomentX	MomentY	Euler	Result
S	254	674	0.00702625	0.624017	0.611432	0.512756	0.513228	1	S
P	238	570	0.00920528	0.65995	0.579849	0.359874	0.441008	0	P
E	278	626	0.00891468	0.621711	0.926222	0.513705	0.405108	1	E
E	273	610	0.00921962	0.579569	0.899699	0.51185	0.413059	1	E
D	253	632	0.00795969	0.680356	0.974429	0.50504	0.46334	0	D
T	155	354	0.015543	0.875162	0.691462	0.29771	0.503452	1	T
M	255	682	0.0068894	0.907905	0.614625	0.550608	0.520765	1	M
I	46	134	0.0321928	0.972104	0.972104	0.671957	0.402826	1	I
I	23	66	0.0663514	0.915048	0.928285	0.473261	0.512391	1	I
L	155	332	0.0176712	0.349477	0.605192	0.723581	0.348032	1	L
5	252	622	0.00818521	0.672788	0.532917	0.49377	0.513413	1	5
9	257	698	0.00662876	0.742794	0.544774	0.442432	0.535895	0	9
1	102	248	0.0208404	0.362832	0.79292	0.337745	0.533039	1	1

Total 13, 13, 100 %

Figure 5. OCR for Test_night. Total character: 13. Correct rate: 100%

Label	Area	Peri	Circle	SymmetrX	SymmetrY	MomentX	MomentY	Euler	Result
D	230	604	0.00792253	0.797459	0.947596	0.498522	0.473783	0	D
E	249	574	0.00949698	0.471617	0.787325	0.490301	0.461747	2	S (*)
E	253	588	0.00919551	0.516618	0.91001	0.510336	0.450059	1	E
P	204	518	0.0095539	0.710613	0.609622	0.382059	0.449363	0	P
S	229	600	0.00799361	0.636758	0.631693	0.49976	0.529891	1	S
T	143	338	0.0157294	0.932781	0.811694	0.292483	0.520944	1	T
I	54	190	0.0187973	0.895272	0.994255	0.287778	0.344815	1	I
M	142	404	0.0109329	0.985058	0.91383	0.539225	0.51493	1	M
I	51	166	0.0232575	0.973622	0.974854	0.731275	0.32049	1	I
L	132	302	0.0181874	0.300885	0.794191	0.716288	0.398636	1	L
5	236	592	0.00846211	0.684593	0.56032	0.49089	0.530424	1	5
2	227	612	0.0076161	0.660681	0.628881	0.522181	0.53663	1	2
V	177	430	0.0120295	0.986771	0.810283	0.602006	0.507655	1	M (*)
R	238	662	0.0068245	0.770102	0.815968	0.435756	0.488193	0	P (*)
Y	125	376	0.0111108	0.952014	0.416478	0.40148	0.51692	1	T (*)
O	224	592	0.00803183	0.937584	0.976613	0.510179	0.515625	0	0 (*)
E	239	550	0.00992847	0.599809	0.973617	0.504372	0.475711	1	E
E	240	560	0.00961712	0.521517	0.850485	0.49175	0.4835	1	E
D	232	600	0.00809833	0.889707	0.970976	0.50125	0.492759	0	D
S	205	566	0.00804139	0.656661	0.650732	0.511049	0.52061	1	S
P	202	520	0.0093876	0.693033	0.691013	0.361139	0.448812	0	P

Total 16, 21, 76 %

Figure 6. OCR for Test_shade. Total character: 21. Correct rate: 76%

1.4 Discussion

1.4.1 Preprocessing Details

Preprocessing methods for Test_ideal1:

Convert to gray image. Convert to binary image. Segment the image to 12 sub-images. Thinning the image.

Preprocessing methods for Test_ideal2:

Convert to gray image. Convert to binary image. Close Operator to the image to smooth the image Segment the image to 12 sub-images. Thinning the image.

Preprocessing methods for Test_night:

Convert to gray image. Histogram equalization. Convert to binary image. Close Operator to the image to smooth the image. Segment the image to 13 sub-images. Thinning the image.

Preprocessing methods for Test_shade:

Convert to gray image. Convert to binary image. Image denoising. Dilate Operator to the image to hole-filling. Segment the image to 21 sub-images. Convert image color white to black, black to white. Dilate Operator to the image to hole-filling. Segment the image to 2 sub-images.

1.4.2 Segmentation

The segmentation method is implemented by counting 8-connectivity pathway and labeling referred from HW3, problem3. As I wrote before, I use two thresholds to avoid too large or too small sub-image. For example, in figure Test_ideal1.raw, the

outline of the sign is not possible to remove become it is clearly and not noisy. If there is no thresholds here, we will can an extra "0" character, the total number of characters in Test_ideal1 will become 13.

Figure 7, Figure 8 descript each step in segment.

Figure 7. Labeling the training image.

Figure 8. Left: Segment one sub-image. Right: Resize it to 100 * 100

1.4.2 Feature Extraction

The result of feature extraction is in figure 1.

Area. This is calculated before thinning operator applying. This is the total number of image which pixel value is not 0.

Perimeter. This is calculated before thinning operator applying. This is the total number of pixel in the edge of the character.

Circularity. Circularity is not being used in this decision tree.

Symmetry. Symmetry includes two directions, x and y axis. This is the probability of pixels in the image that has the same value in the symmetry of certain axis from the total pixels.

Euler Number. This is the number of connected objects subtract the number of holes. Because Euler number is always integer. I use this as the beginning feature in the decision tree.

Spatial Moment. This represent the spatial weight of the image, also includes two directions. My implementation here is use M(0, 1)/M(0, 0) and M(1, 0)/(M(0, 0) first-order result. 0.5 is the most important threshold in first-order spatial moment. 0.5 is the center of the image.

1.4.3 Decision Tree

After developing the decision tree in problem1a, I use the initial decision tree to recognize the testing image. However, I find there are many mistakes in the results**.**

To get a better result, I adjust the decision tree several times to make it more robust. Not only adjusting the perimeter, by testing more images, we can see the importance of different feature to differ each charterer and choose the most robust feature to differ some certain charterers.

Problem 2: Contour Modeling

2.1 Motivation

We will implement two algorithms: the snake algorithm and the level-set algorithm in this problem. Both algorithm is used to detect contour in the gray image. These algorithms are widely used in the field of Biomedical Imaging to detect the contour.

2.2 Algorithm and Implementation

The snake algorithm (Active Contour Model) is an energy-minimizing spline. This algorithm is influenced by lines and edges of the image and also influence by external selection of contour initialization [4]. The code of Snake algorithm is provided by Ritwik Kumar, Snakes: Active Contour Models [7].

The Level-set algorithm is introduced by Osher and Sethian in 1987 which is used to capture the contour of the image. The basic idea of level set is that first choose a Level Set Function (LSF) as a zero-level set of a higher dimensional function, then evolution the level-set function to detect the contour [5]. The code of Level-set algorithm used here is provided by Chunming Li, Level set for image segmentation [6].

2.3 Result

2.3.1 Extract the contour of the spine

The contour results of Spine image are Figure, 9, 10, 11. Figure 9, 10 are Snake algorithm, Figure 11 is Level-Set algorithm.

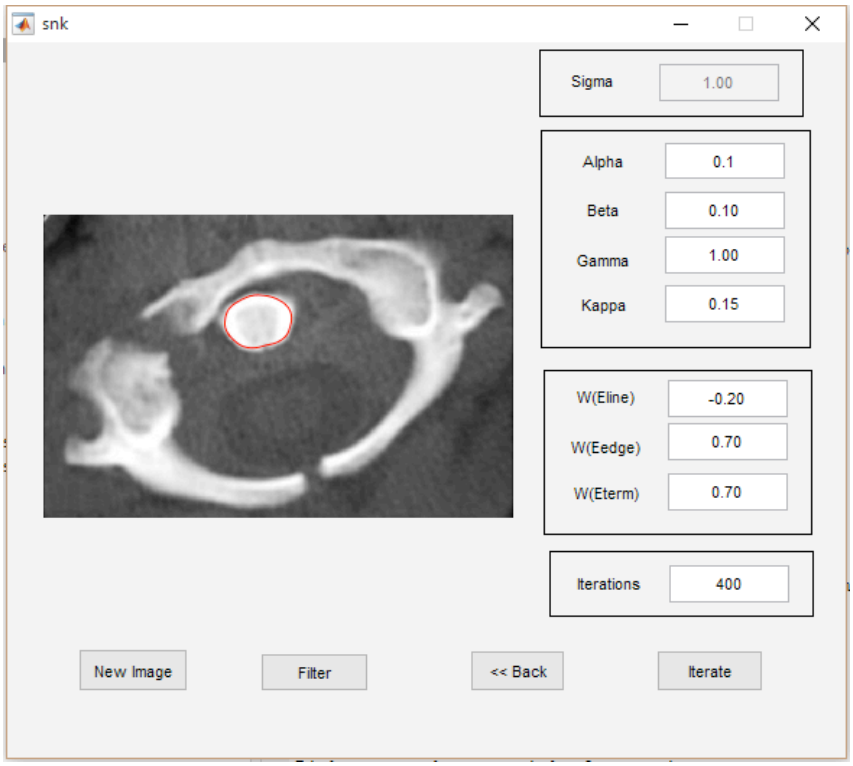

Figure 9. Snake Algorithm of Spine. Alpha = 0.1, Beta = 0.1, Wline = -0.2, Wedge = 0.7, Wterm = 0.7

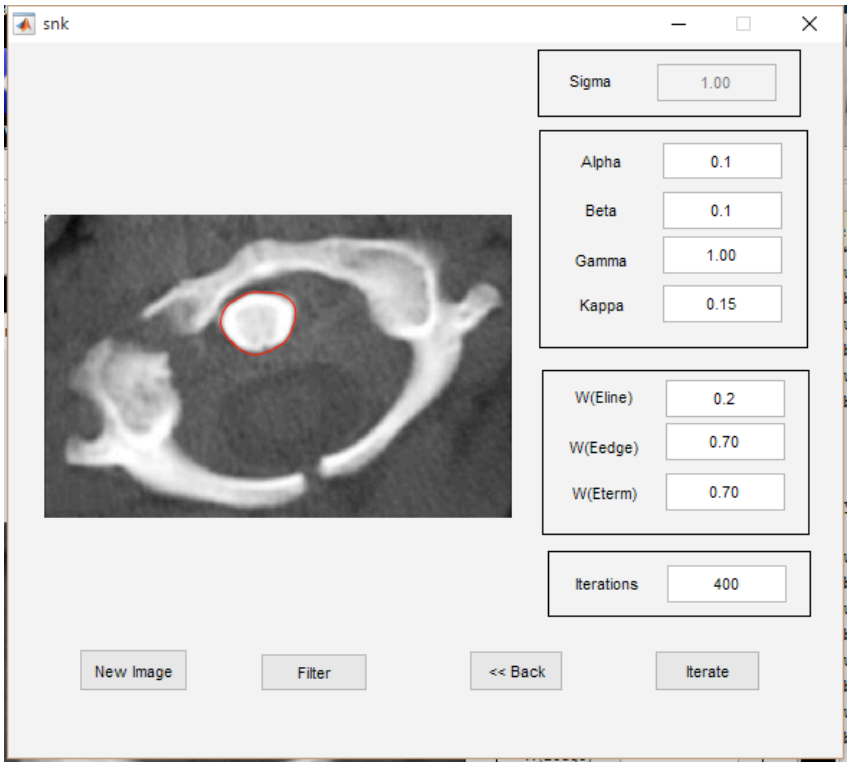

Figure 10. Snake Algorithm of Spine. Alpha = 0.1, Beta = 0.1, Wline = 0.2, Wedge = 0.7, Wterm = 0.7

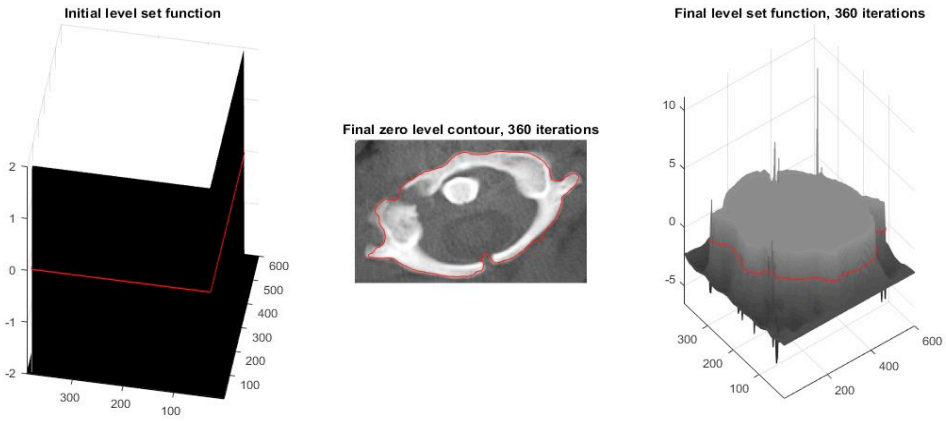

Figure 11. Level-Set Algorithm of Spine. Lambda = 5, Alpha = 2

2.3.2 Extract the outline of coronary from the following angiogram image

The contour results of Coronary image are Figure 12, 13. Figure 12 are Snake algorithm, Figure 13 is Level-Set algorithm.

Figure 12. Snake Algorithm of Coronary. Alpha = 0.4, Beta = 0.4, Wline = 1, Wedge = 5, Wterm = 0.7

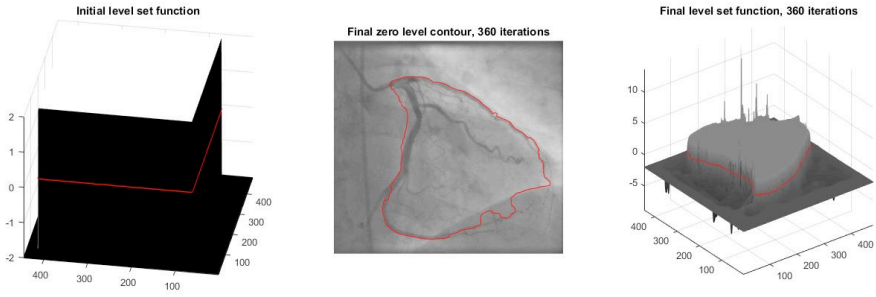

Figure 13. Level-Set Algorithm of Coronary. Lambda = 5, Alpha = 2

2.3.3 Separate red and white blood cells

The contour results of Blood Cell image are Figure 14, 15, 16. Figure 14, 15 are Snake algorithm, Figure 16 is Level-Set algorithm.

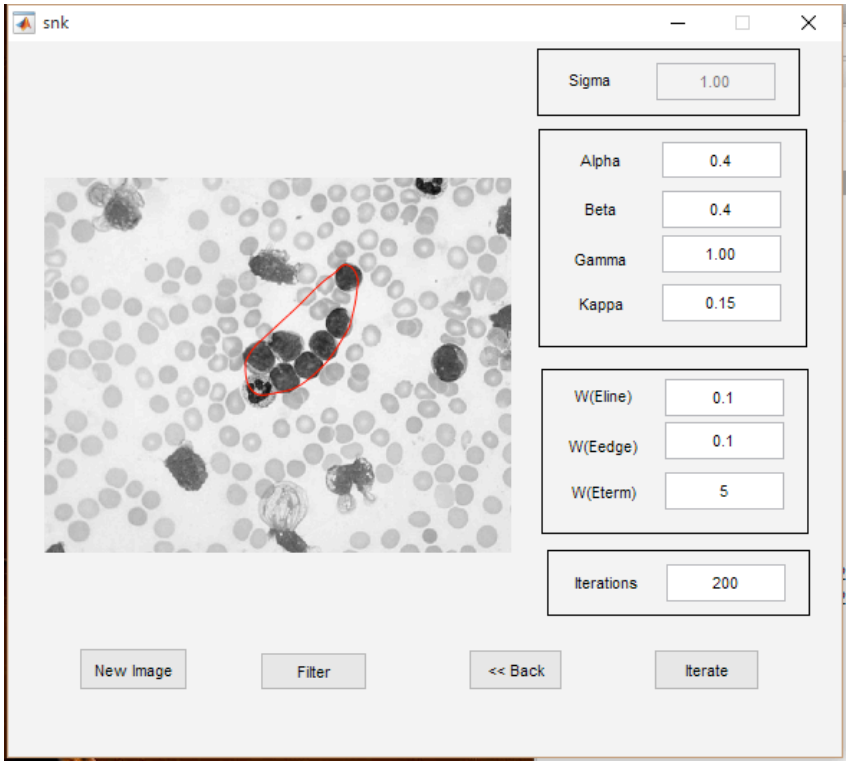

Figure 14. Snake Algorithm of Blood Cell Alpha = 0.4, Beta = 0.4, Wline = 0.1, Wedge = 0.1, Wterm = 5

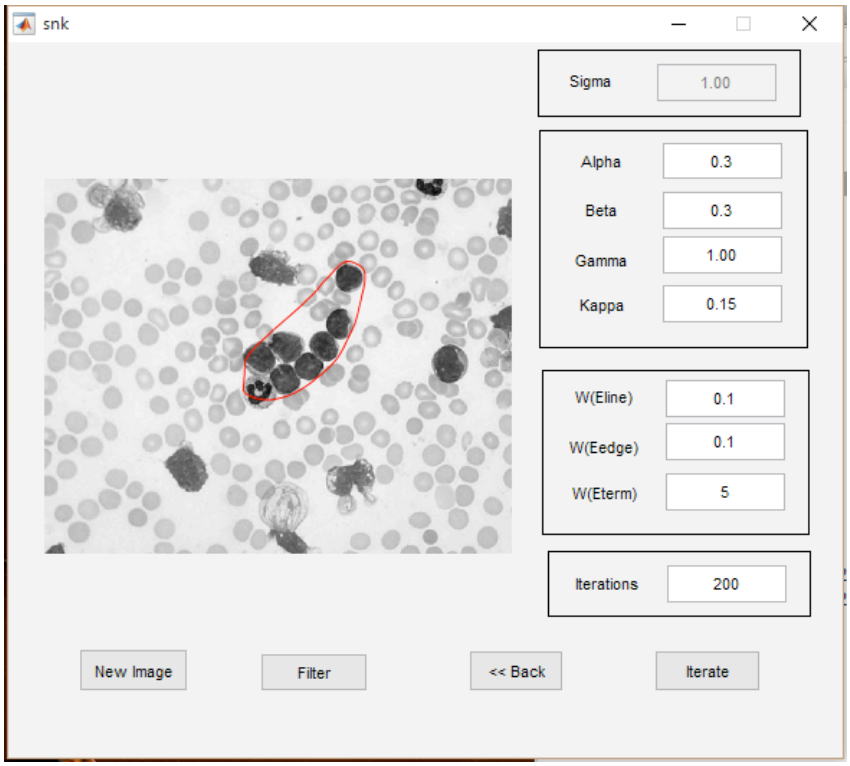

Figure 15. Snake Algorithm of Blood Cell Alpha = 0.3, Beta = 0.3, Wline = 0.1, Wedge = 0.1, Wterm = 5

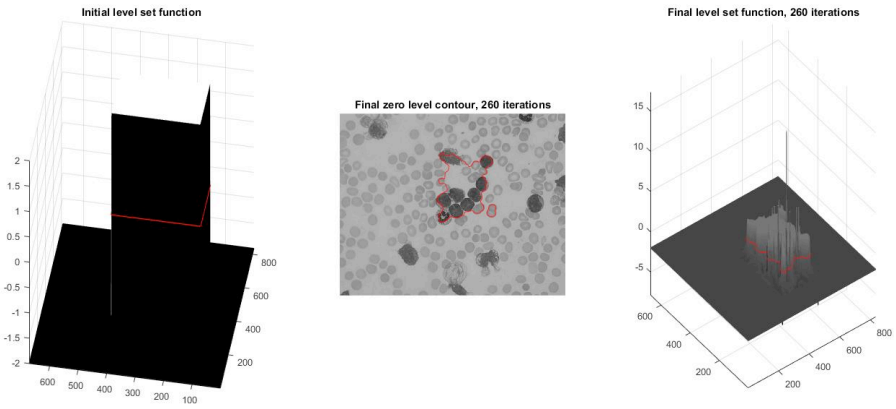

Figure 16. Level-Set Algorithm of Blood Cells. Lambda = 5, Alpha = 2

2.3.4 Isolate the tumor in the brain MIR image

The contour results of Brain image are Figure, 17, 18, 19. Figure 17, 18 are Snake algorithm, Figure 19 is Level-Set algorithm.

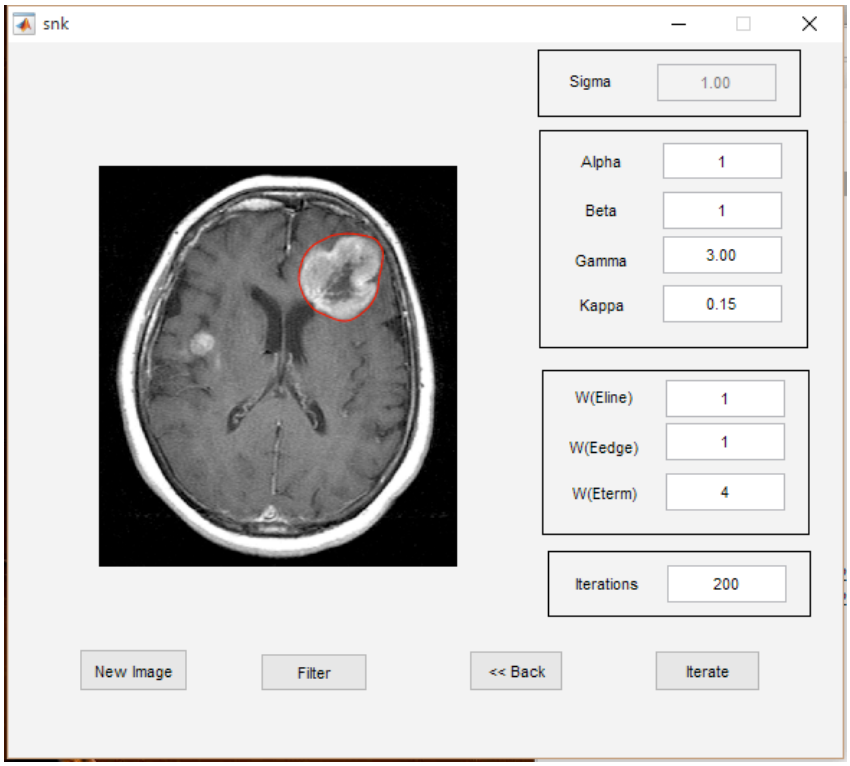

Figure 17. Snake Algorithm of Brain Image. Alpha = 1, Beta = 1, Wline = 1, Wedge = 1, Wterm = 4

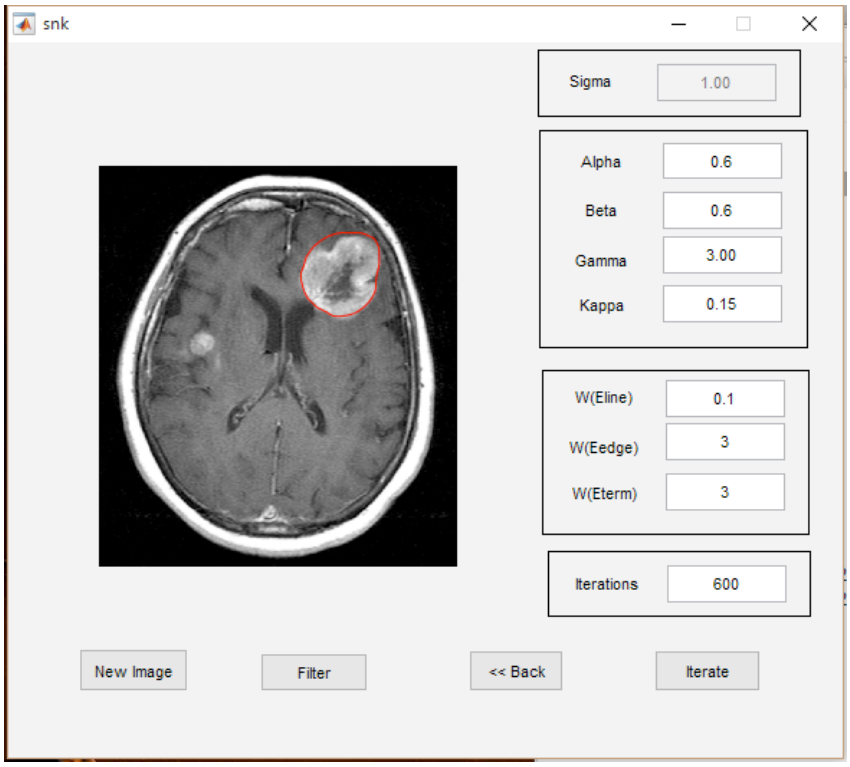

Figure 18. Snake Algorithm of Brain Image. Alpha = 0.6, Beta = 0.6, Wline = 0.1, Wedge = 3, Wterm = 3

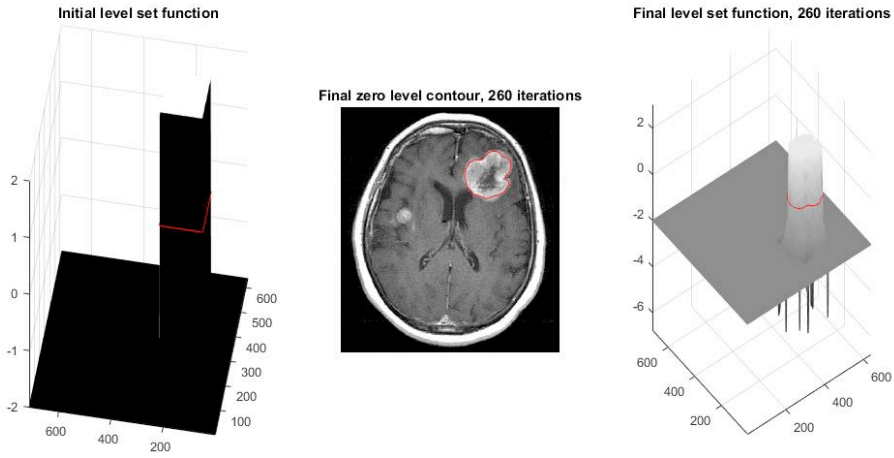

Figure 19. Level-Set Algorithm of Brain Lambda = 5, Alpha = 2

2.4 Discussion

2.4.1 Snake Algorithm

There are several perimeters in Snake Algorithm.

Alpha and beta. Alpha specifies the elasticity of the snake and Beta specifies the rigidity in the contour. For example, snake algorithm of blood cells in Figure 14 and Figure 15, when alpha and beta increase from 0.3 to 0.4, the final contour contract a little bit because the contour become more rigid.

Wline, Wedge, Wterm. Wline is the weighting factor for intensity based potential term. In the first Spine Figure 9 and 10, we can see that the sign of Wline influent the contour direction. Wedge is the weighting factor for edges. For example, in Blood Cell Case Figure 14 and Figure 15, we has to decrease the Wedge to avoid the influence of other white cells. Wterm is the weighting factor or termination potential term. Here I always increase the weight of Wterm if I need to avoid other noise line and want the contour is determined by energy.

The Snake Algorithm depends on the selection of the initialization point. If the number of selected points is too small or too far away from the target, then the result will be not good enough to detect the contour.

2.4.2 Level-Set Algorithm

We can see level-set algorithm mainly depends on three factor: LSF (Level-Set Function), lambda and alpha. The LSF is like the selection point in Snake Algorithm. We can use LSF to adjust the target we want. In the following three images Figure 20, 21,

22, we can that alpha determine the size of the contour, it like the elasticity of the contour, and Lambda influent the sensitive to the edge or line in the image.

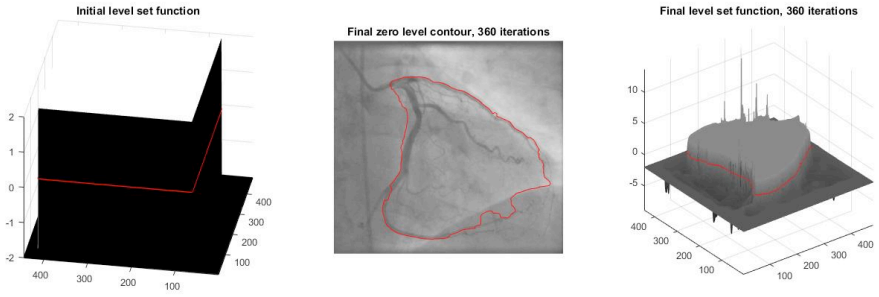

Figure 20. Level-Set Algorithm of Coronary. Lambda = 5, Alpha = 2

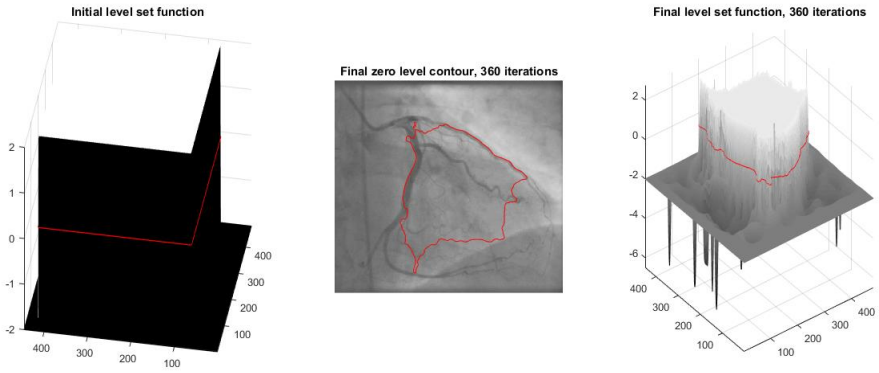

Figure 21. Level-Set Algorithm of Coronary. Lambda =2, Alpha = 2

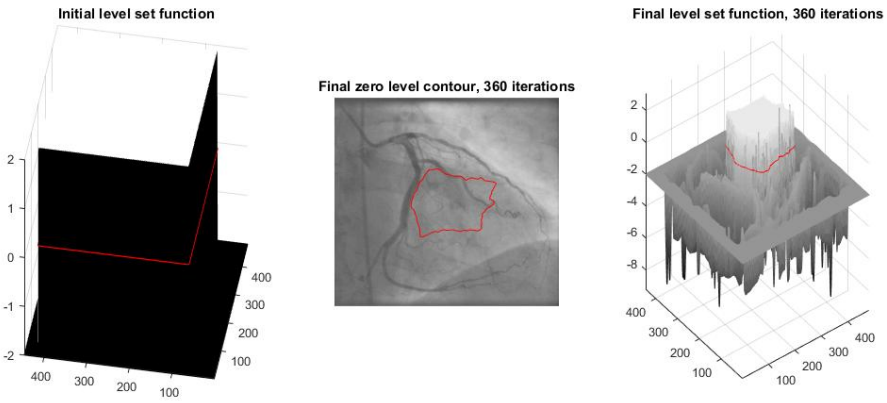

Figure 22. Level-Set Algorithm of Coronary. Lambda =2, Alpha = 5

2.4.3 Compare two methods

Heuristic information. Both Level-set and Snake Algorithm need an external heuristic information. Compared to LSF, the selection of initialization points in Snake Algorithm is easier to adjust. So we can use Snake Algorithm to extract the shape in

the image which we can first select by ourselves.

Shape. Level-Set method perform better in non-round contour image, and Snake Algorithm perform better in round-shape contour image.

Spend time. The level-set spend much more time then Snake Algorithm.

Problem 3. Salient Point Descriptors and Image Matching

3.1 Motivation

This problem is about salient point descriptors extraction and its application: Image Matching. Here SIFT (Scale-Invariant Feature Transform) and SURF (Speeded Up Robust Features) is used to extracted the salient point descriptors.

In problem3b, one-to-one image matching is implemented by SIFT. And in problem3c, we will use SIFT to implement a BoW (Bag of Words) application for one-to-multi image matching.

3.2 Algorithm and Implementation

Scale-Invariant Feature Transform. This is an algorithm in Computer Version to detect and descript the feature. It finds the extrema value point and extract the feature. This algorithm is from Lowe, David in 1999 [8].

Speeded Up Robust Features. SURF is raised by Bay in 2006. SURF is an approximate SIFT algorithm. It works almost equally as the SIFT but spend less time.

Bag of Words. After extracting descriptors via SIFT methods, apply k-means clusters to extract the features and form a codebook. Use the codebook as a dictionary to find the nearest neighbor and match the image.

3.3 Results

3.3.1 Extraction and Desc

Figure 23, 24 is the SIFT results for Bus and Sedan image. Figure 25, 26 is the SURF result.

Figure 23. SIFT for Bus image

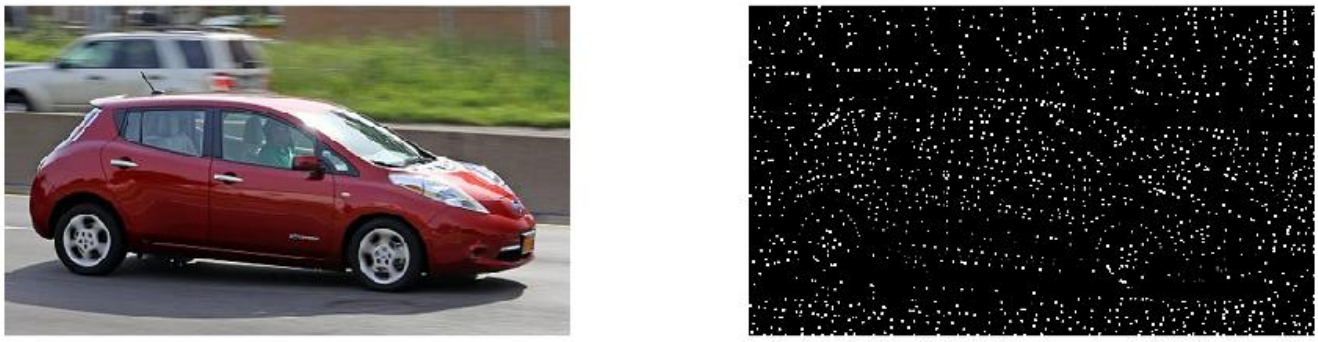

Figure 24. SIFT for Sedan image

Figure 25. SURF for Bus image

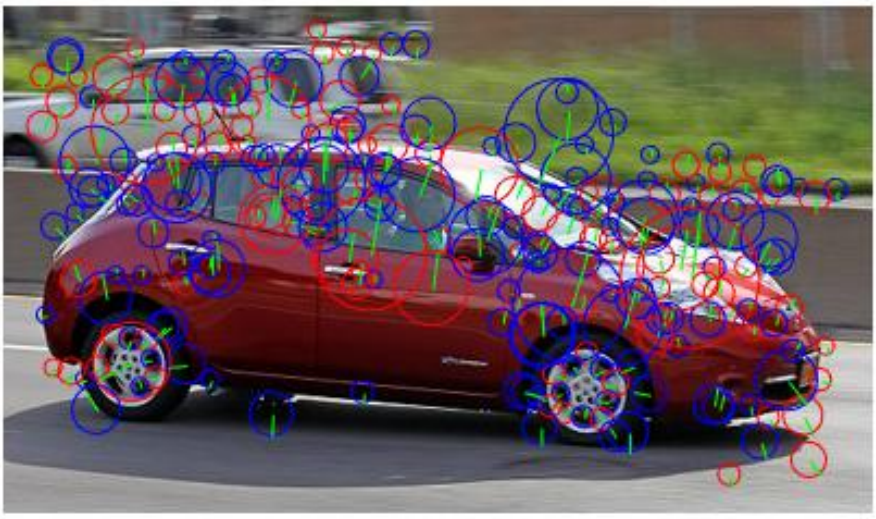

Figure 26. SURF for Sedan image

3.3.2 Image Matching

Figure 27, 28 are the results of extracting feature by SIFT for School_bus1 and School_bus2. Figure 29, 30, 31, 32 is the matching result.

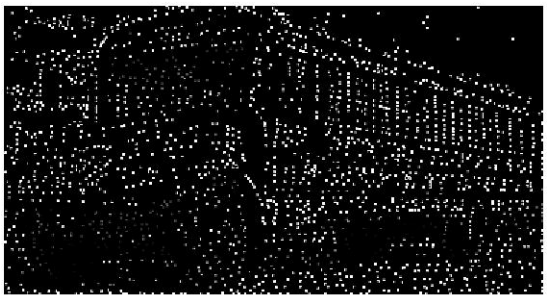

Figure 27. SIFT extraction for School_Bus1

Figure 28. SIFT extraction for School_Bus2

Figure 29. Matching by SIFT for School_bus1 and School_bus2

Figure 30. Matching by SIFT for School_bus1 and Bus

Figure 31. Matching by SIFT for School_bus1 and Sedan

Figure 32. Matching by SIFT for School_bus1 and School_bus1

3.3.3 Bag of Words

Figure 33, 34 are the training and testing results for Bag of Words.

Example: 1, Class: Bus

Example: 1, Class: Sedan

Example: 1, Class: SchoolBus1

Figure 33. Training of image

Figure 34. Testing of images

3.4 Discussion

3.4.1 Compare SIFT and SURF

SURF is an approximation of SIFT. Its advantage spend less time. And by comparing the result of Figure 23,24 and Figure 25,26, we can see that SIFT is too much sensitive to the texture like leafs. If we use SIFT and SURF to match the image, we can assume that SIFT performs better in matching original image and rotated or scaling image. And SURF performs better in more other texture image and classify shape.

3.4.2 Image matching of SIFT

We can see that in Figure 29, and 32, the performance of the image is really good, only one mistake is happened in the head of the School bus, because the texture there is almost black. So the extrama is not very clearly.

However, in Figure 30 and 31, we can see that a lots of mistake occurred. The matching in end of School bus and Sedan is almost opposite. I think the reason is the car of two images are in the opposite direction. SIFT matching perform good in rotation image, but not good enough in symmetrical images.

3.4.3 Bag of Words matching

Bag of Words perform really well in the image matching in this cases.

Comparing to problem3b, BoW can be used more widely, because it can apply a set of images instead of two images. So this can be used in searching an image. One extra information we must provided is the label set of training and testing images.

Reference

[1] https://en.wikipedia.org/wiki/Optical_character_recognition

[2] William K. Pratt "Digital Image Processing"

[3] Shangewen Li "569 Discussion #5

[4] Snakes: Active Contour Models, KASS

[5] Li, Chunming, et al. "Distance regularized level set evolution and its application to image segmentation." *Image Processing, IEEE Transactions on* 19.12 (2010): 3243-3254.

[6] http://www.mathworks.com/matlabcentral/fileexchange/12711-level-set-for-image-segmentation

[7] http://www.mathworks.com/matlabcentral/fileexchange/28109-snakes--active-contour-models