CMPEN/EE455: Digital Image Processing I

Computer Project # 2:

Connected-Component Labeling and Set Operations

Names of Group Members: Jiachen Chen, Yijie Tong, Lyuzhou Zhuang

Date: 09/25/2015

A. Objectives:

- Understanding the concept of threshold and learn to transfer thresholded image.
- Try to use bwlabel and label2rgb functions to transfer images, furthermore, select particular components and save them.
- Understanding how set operations (OR, XOR, NOT) for images work and realizing them in Matlab.
- Learning and implementing min operation between two grey scale images to form a new hybrid image.
-

B. Method:

Problem 1:

In this problem, the first step is to choose a threshold. We set the threshold 127, and set the pixels, whose value is smaller than 127, to 0, and the value larger than 127 to 1. Then we transfer this single image to uint8 image, and then save the image to "fthread". Next, we use the function *bwlabel* to find all 4-connected components. Furthermore, using the function *label2rgb* to display the labeled image.

To save the 3 largest components of the labeled image and delete the other components by setting their constituent pixels to 0. We design the function *select*. In the select function, we first count the size of all components and save them to a 1-D matrix. Then find the 3 largest components, we use a loop showed in following flow chart(Figure 1.1).

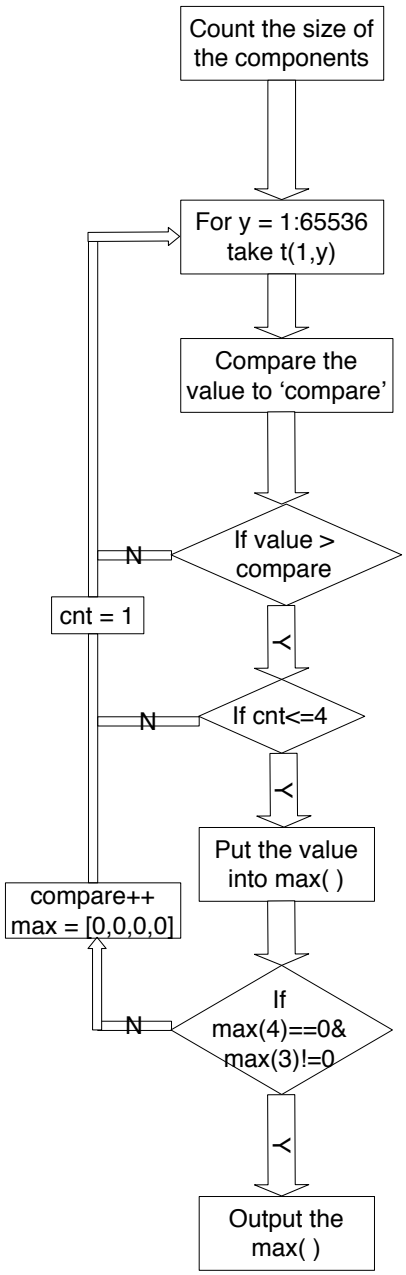

Figure 1.1 Flowchart of looking for the 3 largest components.

After finding the 3 largest three components, we save these 3 components and delete other components to 0. Finally we get the new image.

Problem 2:

In this problem, three logic operators, which are OR, XOR and NOT, need to be constructed. Each corresponding pixel value in two images is compared and set as a new value in the new image matrix according to the truth table shown in Table 1-3.

OR		
A	B	OUTPUT
0	0	0
1	0	1
0	1	1
1	1	1

1

XOR		
A	B	OUTPUT
0	0	0
1	0	1
0	1	1
1	1	0

2

NOT	
INPUT	OUTPUT
0	1
1	0

3

Table 1-3. Truth table for OR; Truth table for XOR; Truth table for NOT.

The second part of the problem requested a function that is equivalent to logic operator AND. It is realized trough constructing a min algorithm that outputs the largest value of two images as a new matrix with the same size.

There are four images used in this part. A and B are binary "bear" and "castle" images respectively. C and D are two grey-scale images called "mandrill_gray" and "cameraman" respectively. The min operator will produce an E image.

Codes for each operation:

```
function [oops]=OR(a,b)
[M,N] = size(a);
for x = 1:M
    for y = 1:N
        if a(x,y)+b(x,y)>=1%compare every pixel values
            oops(x,y)=1; %output 1 if either is 1
        else oops(x,y)=0;
        end
    end
end
```

Code 1.1. Matlab code of OR operation.

```
function [oops]=XOR(a,b)
[M,N] = size(a);
for x = 1:M
    for y = 1:N
        if a(x,y)~=b(x,y)%compare every pixel values
            oops(x,y)=1; %output 1 only if the two values are not the same
        else oops(x,y)=0;
        end
    end
end
```

Code 1.2. Matlab code of XOR operation.

```
function [oops]=NOT(a)
[M,N] = size(a);
for x = 1:M
    for y = 1:N
        if a(x,y)==1%compare every pixel values
            a(x,y)=0; %change 1 to 0
        else
            a(x,y)=1; %change 0 to 1
        end
    end
```

```
end
oops=a;
```

Code 1.3. Matlab code of NOT operation.

```
function [oops]=minimum(a,b)
[M,N] = size(a);
for x = 1:M
    for y = 1:N
        if a(x,y)<=b(x,y) %compare every pixel values
            oops(x,y)=a(x,y); %assign the smaller value to output
        else oops(x,y)=b(x,y);
        end
    end
end
```

Code 1.4. Matlab code of min operation.

C. Result:

Problem 1:

We choose 127 as the threshold, and set the values smaller than 127 to 0， larger than 127 to 1. Then we transfer this single image to uint8 image, and then save the image to “fthread”, shows in figure 2.2

Figure 2.1. Original input image “boat.gif”.

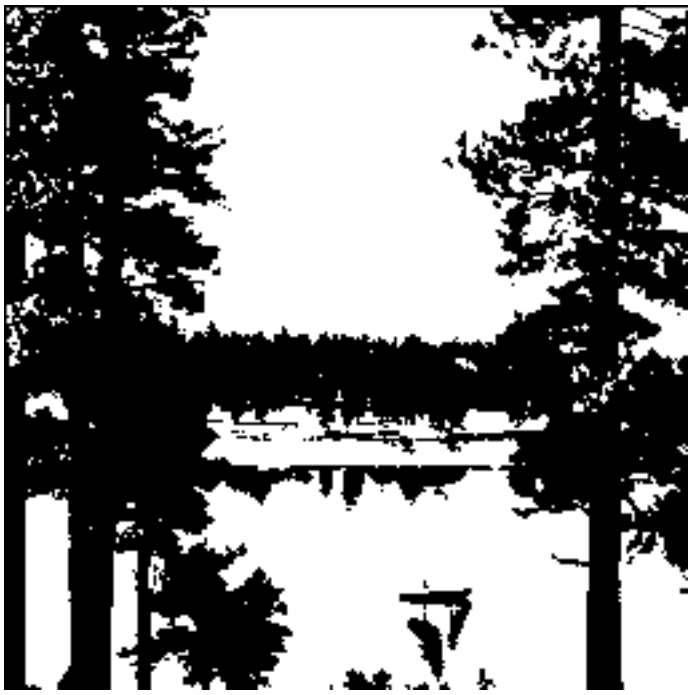

Figure 2.2 Image “fthresh” after threshold the boat image.

Figure 2.3 shows the labeled image with colored components using function fRGB.

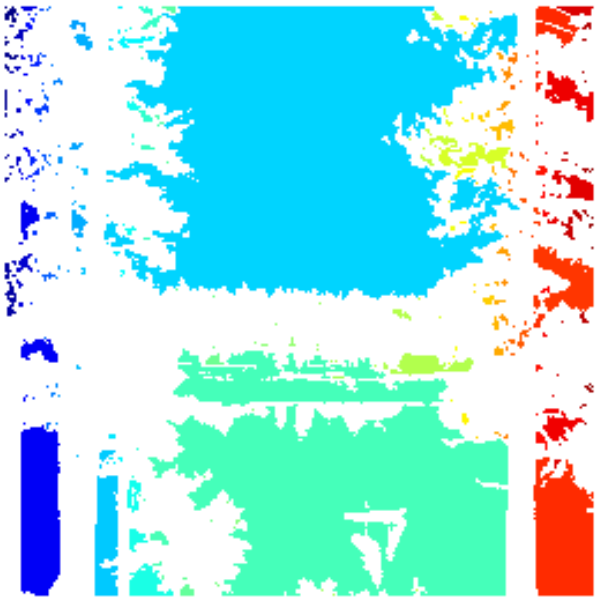

Figure 2.3. Colored image of labeled threshold image.

After running function select, we get the image that just saved the 3 largest components of the labeled image and delete the other components to 0. We can see the result as figure 2.4

Figure 2.4. Image showing selected components

Problem 2:

Original input images for part one are shown in Figure 2.5 and Figure 2.6 and they are stored as matrix A and B.

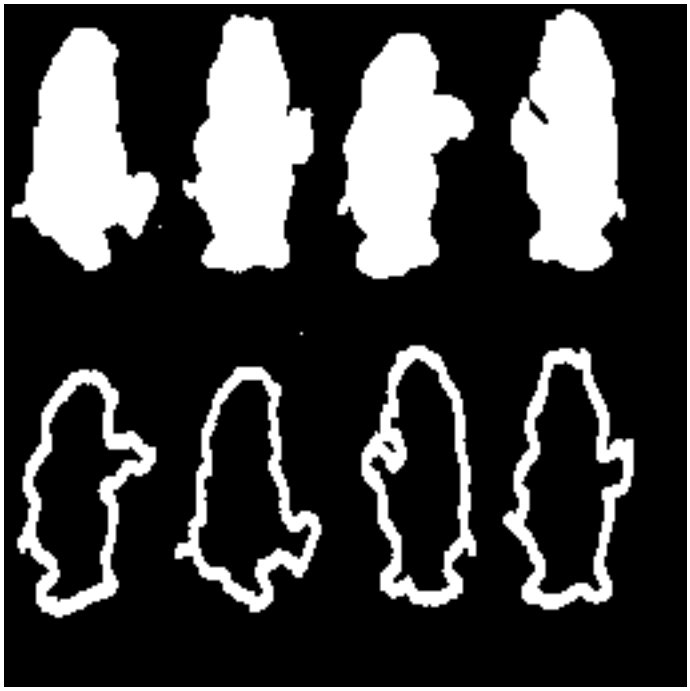

Figure 2.5. Original bear image.

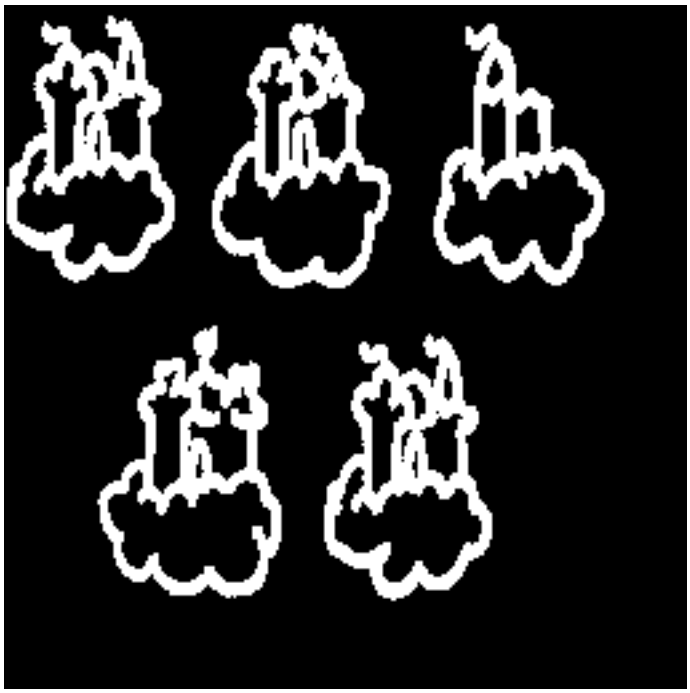

Figure 2.6. Original castle image.

The result of applying OR operation to image A and B is shown in Figure 2.7. The result indicates that the operation overlaps and adds these two images as white edges are blended with the top white bears whereas they are distinguishable at the bottom row with black bears.

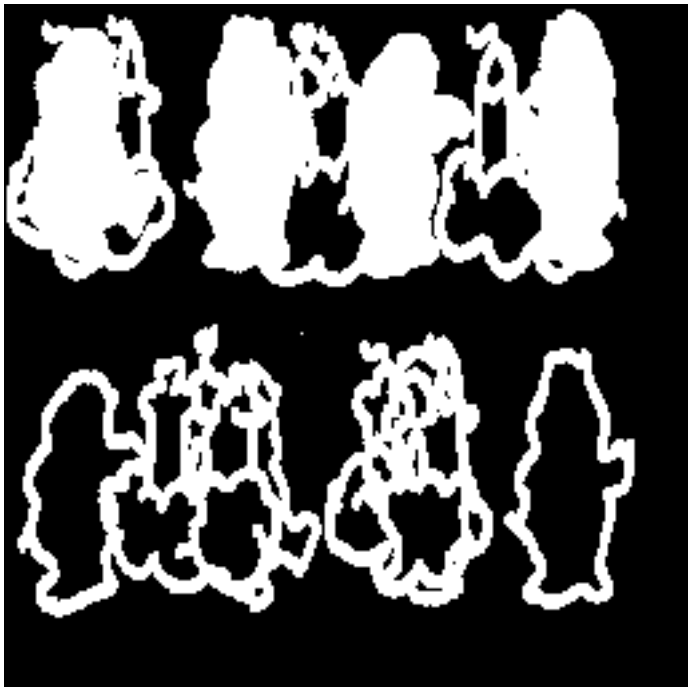

Figure 2.7. Output image of operation “A OR B”.

Figure 2.8 shows the result of XOR operation between image A and B. Unlike the previous operation the contents from two images can be identified as the edges from

two images stay clear. XOR operation can also be expressed as $A \cup B \cap (A \cap B)^C$ where $( )^c$ represents the compliment set.

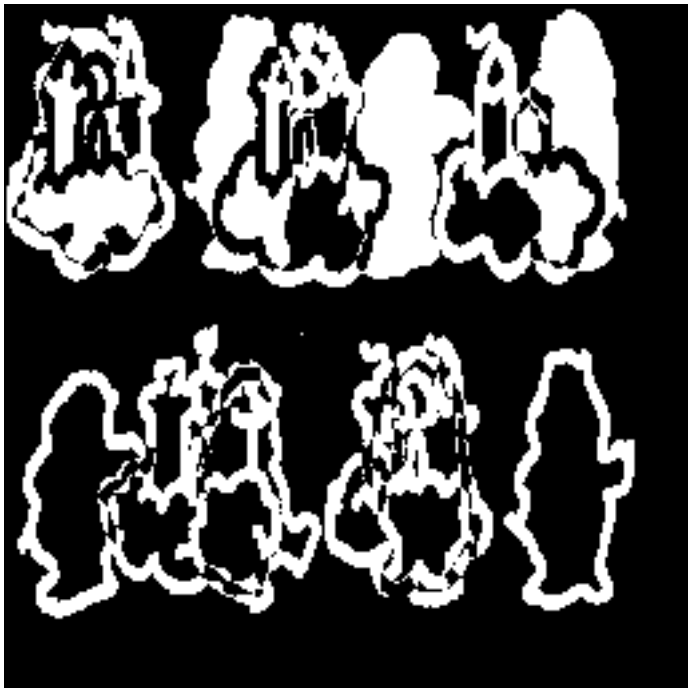

Figure 2.8. Output image of operation “A XOR B”.

Comparing the Figure 2.9 and 2.8, operation NOT(A) XOR NOT(B) gives the same result as operation A XOR B.

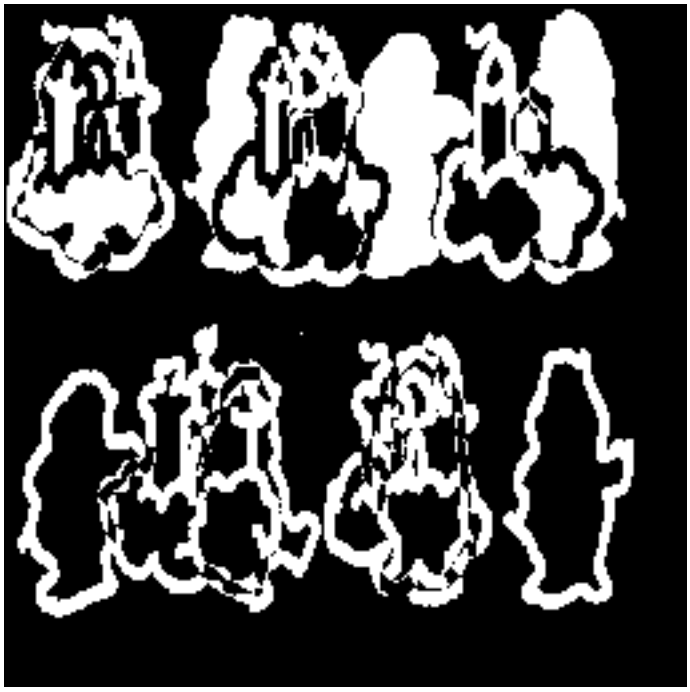

Figure 2.9. Output image of operation “NOT(A) XOR NOT(B)”.

Figure 2.10 and 2.11 show the original images of mandrill and cameraman that are

used for operation min.

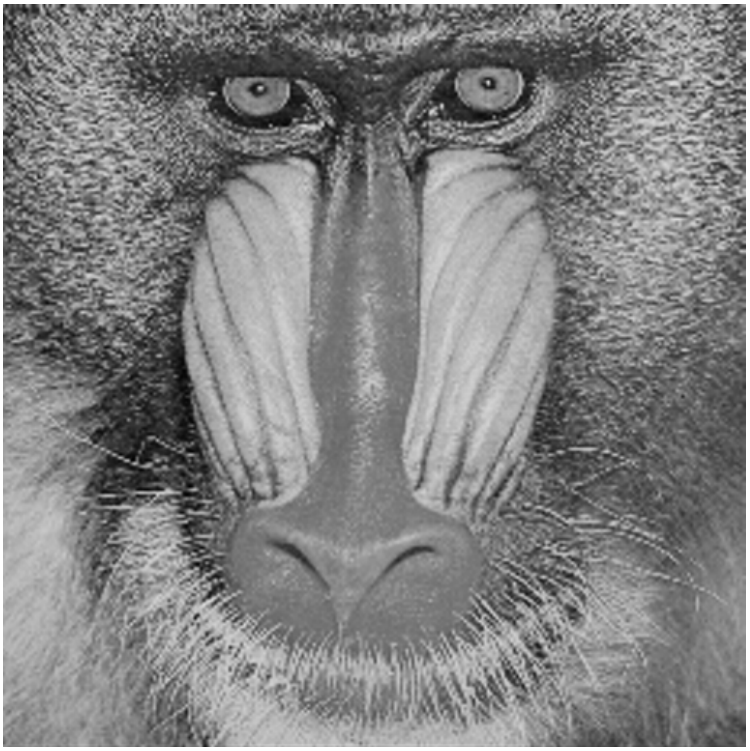

Figure 2.10. Original image of mandrill.

Figure 2.11. Original image of cameraman.

Figure 2.12 is the resultant image after applying min operation to the original images. It is a hybrid image of both images with each pixel values equal to the higher one in either image.

Figure 2.12. Resultant image after min operation between the two original images.

Conclusion:

This project successfully threshold a gray scale image to a binary image and labeled connected components with 4-connectivity. Three largest connected components were selected and displayed; all other components were set to 0 regarded as background. Four basic set operations (OR, XOR, NOT and min) were constructed and tested with images. For binary valued images, NOT(A) XOR NOT(B) and A XOR B produced same results. The operation result from each step was shown.