Practical Assignment №4

Images Morphological Analysis

20321233 zou letian

20321308 cao xinyang

20321309 chen hanzheng

Purpose

1. *Basic morphological operations.* Select an arbitrary image containing shape defects (internal «holes» and external «ledges»). Using basic morphological operations, completely remove or minimize defects.
2. *Objects splitting.* Select an arbitrary binary image containing overlapping objects. Use binary morphology operations to separate objects.
3. *Splitting.* Select an arbitrary image containing a small number of local minima. Perform image segmentation by watershed approach.

Part1 *Basic morphological operations.*

1.1 cv::getStructuringElement(shape, size, anchor),

cv2.getStructuringElement(shape, size, anchor)

— define a structural element with given and size . If the point is not defined then it is placed at the center of kernel.

(a) cv::MORPH_RECT, cv2.MORPH_RECT — a rectangular structuring element with given .

(b) cv::MORPH_CROSS, cv2.MORPH_CROSS — a cross-shaped structuring element with given .

(c) cv::MORPH_ELLIPSE, cv2.MORPH_ELLIPSE — an elliptic structuring element inscribed into a rectangle with given

.

1.2 cv::morphologyEx(A, C, op, B, anchor, iters, borderType, borderValue), cv::morphologyEx(A, op, B, anchor, iters, borderType, borderValue)

— apply a morphological filtering operation to an image with structural element times. If the number of iterations is not defined, then it is defaulted to 1. It also allows redefining the point value and function behavior near borders. In C++ language the resulting image is passed as a second argument, while in Python language it is a return value of a function. The possible morphological operations are:

(a) cv::MORPH_ERODE, cv2.MORPH_ERODE — perform the mor

phological erosion operation.

cv::erode() (Python

cv2.erode()) function can be used to do the same oper

ation.

(b) cv::MORPH_DILATE, cv2.MORPH_DILATE — perform the mor

phological dilation operation.

cv::dilate() (Python

cv2.dilate()) function can be used to do the same op

eration. () = ((),).

(c) cv::MORPH_OPEN, cv2.MORPH_OPEN — perform the morpho

logical opening operation.

(d) cv::MORPH_CLOSE,

cv2.MORPH_CLOSE

— perform the morphological closing operation.

(e) cv::MORPH_GRADIENT, cv2.MORPH_GRADIENT — perform the

morphological gradient operation. This is the difference be

tween dilation and erosion.

(f) cv::MORPH_TOPHAT, cv2.MORPH_TOPHAT — perform the mor

phological «top hat» operation. This is the difference

between image and its opening.

(g) cv::MORPH_BLACKHAT, cv2.MORPH_BLACKHAT — perform the morphological «black hat» operation. This is the differ ence between image closing and an image.

(h) cv::MORPH_HITMISS, cv2.MORPH_HITMISS — perform the opening «hit & miss» operation.

1.3 cv::connectedComponents(A, labels, connectivity), cv2.connectedComponents

— lists all connected components of the image , labels every image pixel with corresponding label and stores it it in matrix. It returns the number of connected components (for C++) or a tuple with number of components and a matrix with component (for Python).

1.4 cv::connectedComponentsWithStats(A, labels, stats, centers), cv2.connectedComponentsWithStats(A)

— the same as above () function, but also computes statistics for each of the components. It returns the number of connected components (for C++) or a tuple with number of components and a matrix with component , and (for Python). Statistics () is a 2-dimensional matrix with first dimension being a component id () and the second dimension being a statistic id. Type of this matrix is 32 bit signed integer . Component is a 2-dimensional matrix with first dimension being a component id () and the second dimension being 0 for , and 1 for . Statistics include the following data:

(a) cv::CC_STAT_LEFT, cv2.CC_STAT_LEFT — leftmost () coordinate of a component;

(b) cv::CC_STAT_TOP, cv2.CC_STAT_TOP — topmost () coordinate of a component;

(c) cv::CC_STAT_WIDTH, cv2.CC_STAT_WIDTH — horizontal size (width) of a component;

(d) cv::CC_STAT_HEIGHT, cv2.CC_STAT_HEIGHT — vertical size (height) of a component;

(e) cv::CC_STAT_AREA, cv2.CC_STAT_AREA — component size (in pixels).

MATLAB 's bwareaopen(A, dim) can be easily implemented using OpenCV 's connectedComponentsWithStats(). Please refer to Appendix 4.1 for example source codes with OpenCV and C++ or Python programming languages.

2. Original images;

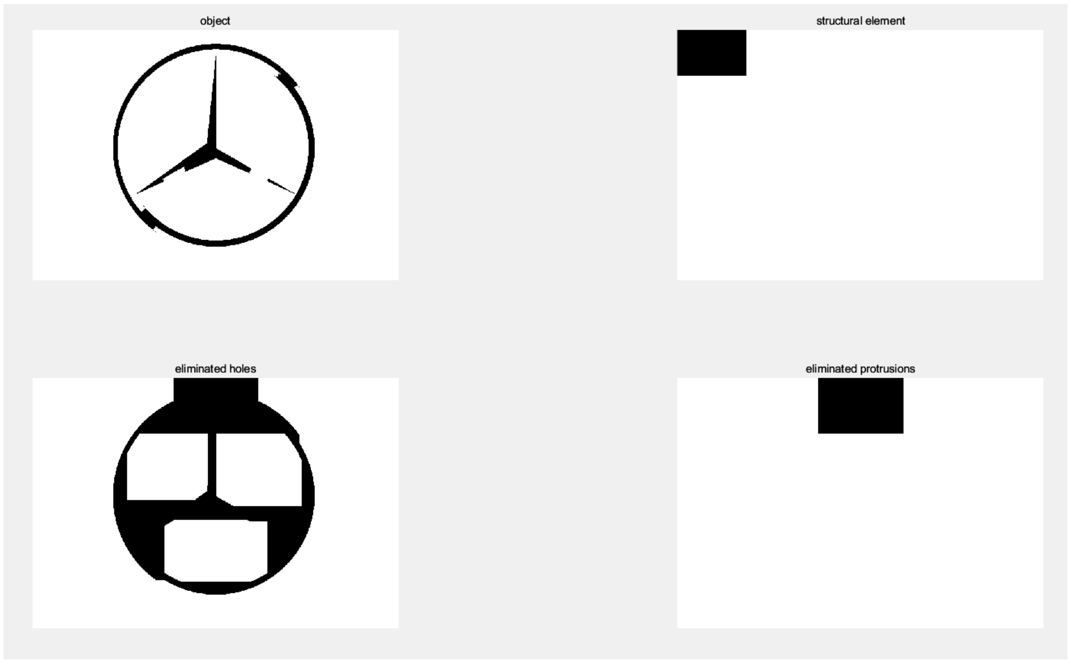
3. Code of the scripts;

```
clc
clear
figure

%%1.FILTERING

I = imread('R-C.jpg');
I = im2gray(I);
t = graythresh(I);
A = imbinarize(I,t);
[numRows,numCols,Layers] = size(A);
    subplot(2,2,1);
imshow(A);
title('object');

B = strel('rectangle',[100 150]);

M = B.Neighborhood;
M(numRows,numCols) = 0;
subplot(2,2,2);
imshow(~M);
title('structural element');

Co = imopen(A,B);
subplot(2,2,3);
imshow(Co);
title('eliminated holes');

Cc = imclose(Co,B);
subplot(2,2,4);
imshow(Cc);
title('eliminated protrusions');
```

4. Comments;

Read the picture and use the built-in imbinarize, strel, subplot, imopen, imclose and other functions of MATLAB to process and generate the corresponding image.

5. Resulting images.



Part2 *Objects splitting.*

2. Original images;

3. Code of the scripts;

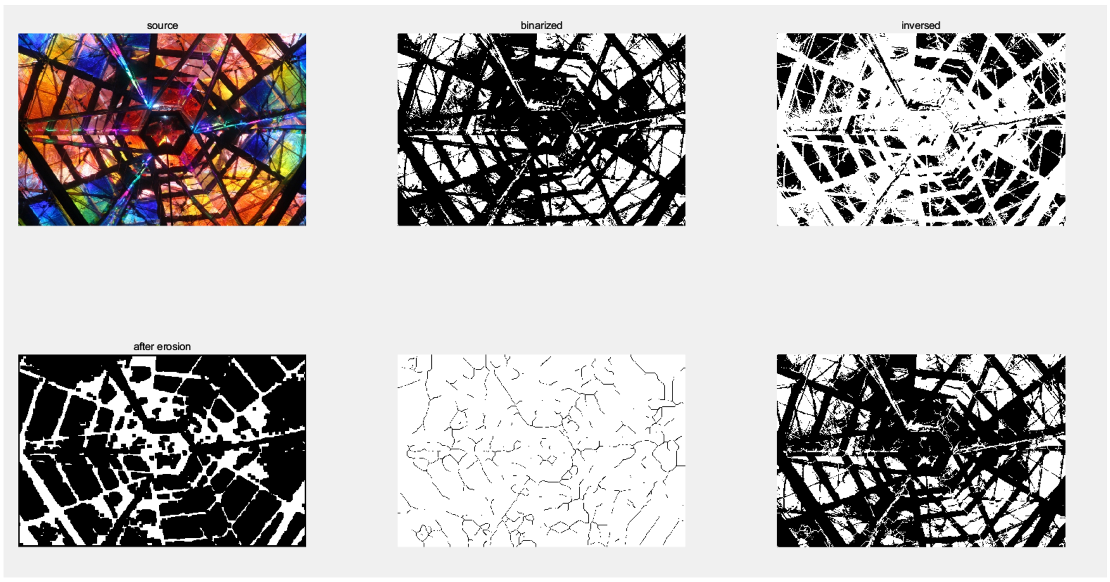
```
%%2.SPLITTING
```

```
I = imread('wht.jpg');
subplot(2,3,1);
imshow(I);
title('source');

I = rgb2gray(I);
t = graythresh(I);
Inew = imbinarize(I,t);
subplot(2,3,2);
imshow(Inew);
title('binarized');

Inew = ~Inew;
subplot(2,3,3);
imshow(Inew);
title('inversed');

BW2 = bwmorph(Inew,'erode',5);
subplot(2,3,4);
imshow(BW2);
title('after erosion');

BW2 = bwmorph(BW2,'thicken',Inf);
subplot(2,3,5);
imshow(BW2);

Inew = ~(Inew & BW2);
subplot(2,3,6);
imshow(Inew);
```

4. Comments;

Read the image and use the built-in imbinarize, subplot, bwmorph and other functions of MATLAB to process it and generate the corresponding picture.

5. Resulting images.



Part3 *Splitting.*

2. Original images;

4. Code of the scripts;

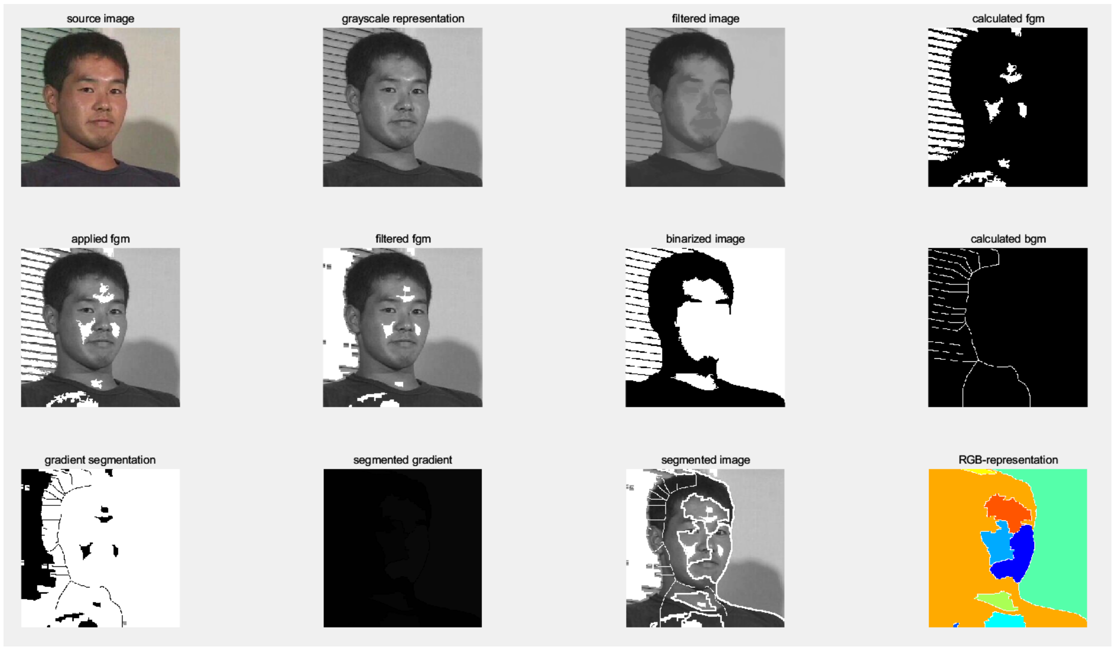
```
%%3.WATERSHED SEGMENTATION

rgb = imread('yjsp.jpg');
subplot(3,4,1);
imshow(rgb);
title('source image');

A = rgb2gray(rgb);
subplot(3,4,2);
imshow(A);
title('grayscale representation');

B = strel('disk',6);
C = imerode(A,B);
Cr = imreconstruct(C,A);
Crd = imdilate(Cr,B);
Crdr = imreconstruct(imcomplement(Crd),imcomplement(Cr));
Crdr = imcomplement(Crdr);
subplot(3,4,3);
imshow(Crdr);
title('filtered image');

fgm = imregionalmax(Crdr);
subplot(3,4,4);
imshow(fgm);
title('calculated fgm');

A2 = A;
A2(fgm) = 255;
subplot(3,4,5);
imshow(A2);
title('applied fgm');

B2 = strel(ones(5,5));
fgm = imclose(fgm,B2);
fgm = imerode(fgm,B2);
fgm = bwareaopen(fgm,20);
A3 = A;
A3(fgm) = 255;
subplot(3,4,6);
imshow(A3);
```

```
title('filtered fgm');

bw = imbinarize(Crdr);
subplot(3,4,7);
imshow(bw);
title('binarized image');

D = bwdist(bw);
L = watershed(D);
bgm = L == 0;
subplot(3,4,8);
imshow(bgm);
title('calculated bgm');

hy = fspecial('sobel');
hx = hy';
Ay = imfilter(double(A),hy,'replicate');
Ax = imfilter(double(A),hx,'replicate');
grad = sqrt(Ax.^2 + Ay.^2);
grad = imimposemin(grad,bgm |fgm);
subplot(3,4,9);
imshow(grad);
title('gradient segmentation');

L = watershed(grad);
subplot(3,4,10);
imshow(L);
title('segmented gradient');

A4 = A;
A4(imdilate(L == 0,ones(3,3)) | bgm |fgm) = 255;
subplot(3,4,11);
imshow(A4);
title('segmented image');

Lrgb = label2rgb(L,'jet','w','shuffle');
subplot(3,4,12);
imshow(Lrgb);
title('RGB-representation');
```

4. Comments;

Read the image and use the corresponding function built into MATLAB to process it and generate the corresponding image.

5. Resulting images.



Conclusion

We have a better understanding of morphology. Morpho logy has basic operations, including expansion and e rosion, and more advanced, open and closed operation s based on the first two. I learned how to use t hese morphological operations to extract the borders of objects and eliminate the protrusions and other defects of objects. All in all, we can now do m uch more in-depth processing of images at the code level.

Questions to Practical Assignment Report Defense

1. Does the opening result include the closing result?

- No. Open results in the body, close results in the outside.
-

2. What morphological filter should be applied to remove ledges from an object?

Erosion.

3. How can you find the object edges using morphological operations?

1. Image A is corroded.
2. The extracted boundary is the result of A minus corrosion

4. What is *morphology*?

Morphology is mainly used to extract meaningful image components from images to express and describe regional shapes so that subsequent recognition can capture the most essential shape features of target objects.