

## Homework 4 for Kun

Introduce to image process

All codes are attached on the last page.

### Edge Detection:

```
1  % Edge Detection
2  I = imread('lena_std.tif');
3  imshow(I), title('Original Image');
4
5  % Roberts operator
6  filter1 = edge(I, 'Roberts');
7  imshow(filter1), title('Roberts operator');
8
9  % Canny operator
10 filter2 = edge(I, 'Canny');
11 imshow(filter2), title('Canny operator');
12
13 % Sobel operator
14 filter3 = edge(I, 'Sobel');
15 imshow(filter3), title('Sobel operator');
16
17 % Prewitt operator
18 filter4 = edge(I, 'Prewitt');
19 imshow(filter4), title('Prewitt operator');
```

Res:

Original Image



Roberts operator



Canny operator



## Sobel operator



## Prewitt operator



I think Sobel Operator gives the best performance, because

---

## Edge Filter

```
21 % Edge Filter
22 o = gaussianOperator(7,1);
23 fil_gau = imfilter(I,o);
24 imshow(fil_gau), title("After Sobel-ish operator");
```

```
113 function o=gaussianOperator(size,sigma)
114 o = zeros(size,size);
115 for i=-(size-1)/2:(size-1)/2
116     for j=-(size-1)/2:(size-1)/2
117         x0=(size+1)/2;
118         y0=(size+1)/2;
119         x=i+x0;
120         y=j+y0;
121         o(y,x)=(x)/(-2*pi*((sigma)^4))*exp(-(x-x0)^2+(y-y0)^2)/2/sigma/sigma);
122     end
123 end
124 sum1=sum(o);
125 sum2=sum(sum1);
126 o=o/sum2;
127 end
```

Res:

After Sobel-ish operator



Operator: The weight for left and right obey for Gaussian rule, more like a gaussian filter.

Image: It seems like more blurred the edge.

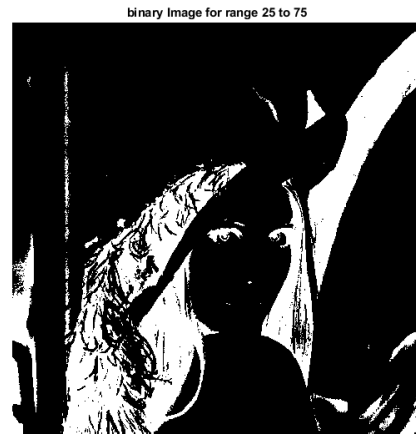
---

## Histogram-based segmentation

a

```
26 % Histogram-based segmentation
27 % a
28 imshow(I), title("Original Image");
```

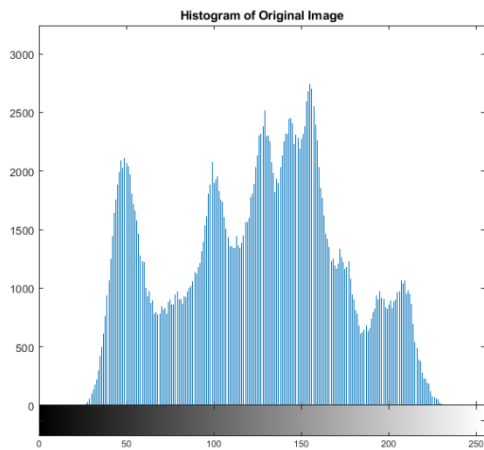
Res



B

```
30 % b
31 imhist(I), title("Histogram of Original Image");
```

Res



C

```
33 % c
34 R1 = [25,75];
35 R2 = [160,180];
36 R3 = [200,300];
```

D

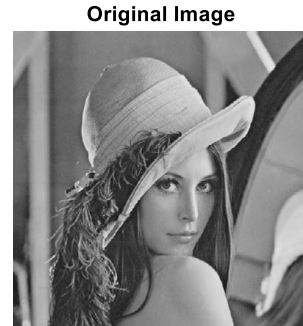
```
38 % d
39 B1 = uint8(255*((I>=R1(1))&(I<=R1(2))));
40 imshow(B1), title("binary Image for range "+R1(1)+" to "+R1(2));
41 B2 = uint8(255*((I>=R2(1))&(I<=R2(2))));
42 imshow(B2), title("binary Image for range "+R2(1)+" to "+R2(2));
43 B3 = uint8(255*((I>=R3(1))&(I<=R3(2))));
44 imshow(B3), title("binary Image for range "+R3(1)+" to "+R3(2));
```

Res

E

```
46 %e
47 base = sum(sum(B1))+sum(sum(B2));
48 m1 = sum(sum(B1.*I))/base;
49 m2 = sum(sum(B2.*I))/base;
50 c = m1*B1 + m2*B2;
51 imshow(c), title("Histogram Segmented Image");
```

Res



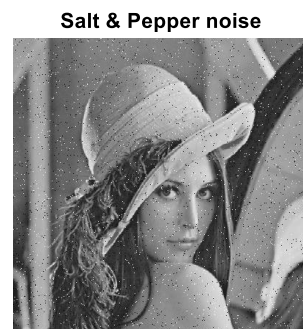
## Noise Reduction

a

```
26 % Noise reduction
27 % a
28 noise_gaussian = imnoise(I,'gaussian',0,0.05);
29 noise_saltAndpepper = imnoise(I,'salt & pepper',0.02);
```

b

```
31 % b
32 figure;
33 subplot(3,1,1);
34 imshow(I), title("Original Image");
35 subplot(3,1,2);
36 imshow(noise_gaussian), title("Gaussian noise");
37 subplot(3,1,3);
38 imshow(noise_saltAndpepper), title("Salt & Pepper noise");
```



Res:

---

C

```

40 % c
41 h3 = fspecial('average',3);
42 h5 = fspecial('average',5);
43 h7 = fspecial('average',7);
44 fil_gau3 = imfilter(noise_gaussian,h3);
45 fil_gau5 = imfilter(noise_gaussian,h5);
46 fil_gau7 = imfilter(noise_gaussian,h7);
47
48 figure;
49 subplot(4,1,1);
50 imshow(noise_gaussian), title("Gaussian noise");
51 subplot(4,1,2);
52 imshow(fil_gau3), title("after 3x3 average filter");
53 subplot(4,1,3);
54 imshow(fil_gau5), title("after 5x5 average filter");
55 subplot(4,1,4);
56 imshow(fil_gau7), title("after 7x7 average filter");
57
58 fil_slt3 = imfilter(noise_saltAndpepper,h3);
59 fil_slt5 = imfilter(noise_saltAndpepper,h5);
60 fil_slt7 = imfilter(noise_saltAndpepper,h7);
61
62 figure;
63 subplot(4,1,1);
64 imshow(noise_saltAndpepper), title("Salt & Pepper noise");
65 subplot(4,1,2);
66 imshow(fil_slt3), title("after 3x3 average filter");
67 subplot(4,1,3);
68 imshow(fil_slt5), title("after 5x5 average filter");
69 subplot(4,1,4);
70 imshow(fil_slt7), title("after 7x7 average filter");

```

Res:

Gaussian noise



after 3x3 average filter



after 5x5 average filter



after 7x7 average filter



D

**Salt & Pepper noise**



```
72 % d
73 m3 = medfilt2(noise_saltAndpepper,[3 3]);
74 m5 = medfilt2(noise_saltAndpepper,[5 5]);
75 m7 = medfilt2(noise_saltAndpepper,[7 7]);
76
77 figure;
78 subplot(4,1,1);
79 imshow(noise_saltAndpepper), title("Salt & Pepper noise");
80 subplot(4,1,2);
81 imshow(m3), title("after 3x3 median filter");
82 subplot(4,1,3);
83 imshow(m5), title("after 5x5 median filter");
84 subplot(4,1,4);
85 imshow(m7), title("after 7x7 median filter");
```

Res:

**after 3x3 average filter**



**after 5x5 average filter**



**after 7x7 average filter**



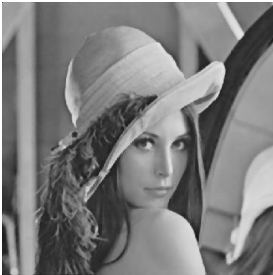
**Salt & Pepper noise**



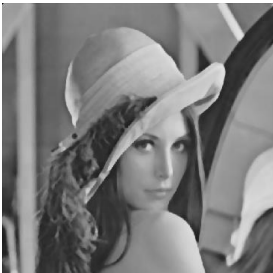
**after 3x3 median filter**



**after 5x5 median filter**



**after 7x7 median filter**





```

% Edge Detection
I = im2uint8(rgb2gray(imread("lena_std.tif")));
imshow(I), title("Original Image");

% Roberts operator
filter1 = edge(I,"Roberts");
imshow(filter1), title("Roberts operator");

% Canny operator
filter2 = edge(I,"Canny");
imshow(filter2), title("Canny operator");

% Sobel operator
filter3 = edge(I,"Sobel");
imshow(filter3), title("Sobel operator");

% Prewitt operator
filter4 = edge(I,"Prewitt");
imshow(filter4), title("Prewitt operator");

% Edge Filter
o = gaussianOperator(7,1);
fil_gau = imfilter(I,o);
imshow(fil_gau), title("After Sobel-ish operator");

% Histogram-based segmentation
% a
imshow(I), title("Original Image");

% b
imhist(I), title("Histogram of Original Image");

% c
R1 = [25,75];
R2 = [160,180];
R3 = [200,300];

% d
B1 = uint8(255*((I>=R1(1))&(I<=R1(2))));

```



```

imshow(B1), title("binary Image for range "+R1(1)+" to "+R1(2));
B2 = uint8(255*((I>=R2(1))&(I<=R2(2))));
imshow(B2), title("binary Image for range "+R2(1)+" to "+R2(2));
B3 = uint8(255*((I>=R3(1))&(I<=R3(2))));
imshow(B3), title("binary Image for range "+R3(1)+" to "+R3(2));

%e
base = sum(sum(B1))+sum(sum(B2));
m1 = sum(sum(B1.*I))/base;
m2 = sum(sum(B2.*I))/base;
c = m1*B1 + m2*B2;
imshow(c), title("Histogram Segmented Image");

% Noise reduction
% a
noise_gaussian = imnoise(I,'gaussian',0,0.05);
noise_saltAndpepper = imnoise(I,'salt & pepper',0.02);

% b
figure;
subplot(3,1,1);
imshow(I), title("Original Image");
subplot(3,1,2);
imshow(noise_gaussian), title("Gaussian noise");
subplot(3,1,3);
imshow(noise_saltAndpepper), title("Salt & Pepper noise");

% c
h3 = fspecial('average',3);
h5 = fspecial('average',5);
h7 = fspecial('average',7);
fil_gau3 = imfilter(noise_gaussian,h3);
fil_gau5 = imfilter(noise_gaussian,h5);
fil_gau7 = imfilter(noise_gaussian,h7);

figure;
subplot(4,1,1);
imshow(noise_gaussian), title("Gaussian noise");
subplot(4,1,2);
imshow(fil_gau3), title("after 3x3 average filter");
subplot(4,1,3);
imshow(fil_gau5), title("after 5x5 average filter");
subplot(4,1,4);
imshow(fil_gau7), title("after 7x7 average filter");

```

```

fil_slt3 = imfilter(noise_saltAndpepper,h3);
fil_slt5 = imfilter(noise_saltAndpepper,h5);
fil_slt7 = imfilter(noise_saltAndpepper,h7);

figure;
subplot(4,1,1);
imshow(noise_saltAndpepper), title("Salt & Pepper noise");
subplot(4,1,2);
imshow(fil_slt3), title("after 3x3 average filter");
subplot(4,1,3);
imshow(fil_slt5), title("after 5x5 average filter");
subplot(4,1,4);
imshow(fil_slt7), title("after 7x7 average filter");

% d
m3 = medfilt2(noise_saltAndpepper,[3 3]);
m5 = medfilt2(noise_saltAndpepper,[5 5]);
m7 = medfilt2(noise_saltAndpepper,[7 7]);

```

```

figure;
subplot(4,1,1);
imshow(noise_saltAndpepper), title("Salt & Pepper noise");
subplot(4,1,2);
imshow(m3), title("after 3x3 median filter");
subplot(4,1,3);
imshow(m5), title("after 5x5 median filter");
subplot(4,1,4);
imshow(m7), title("after 7x7 median filter");

```

```

function o=gaussianOperator(size,sigma)
    o = zeros(size,size);
    for i=-(size-1)/2:(size-1)/2
        for j=-(size-1)/2:(size-1)/2
            x0=(size+1)/2;
            y0=(size+1)/2;
            x=i+x0;
            y=j+y0;
            o(y,x)=(x)/(-2*pi*((sigma)^4))*exp(-((x-x0)^2+(y-
y0)^2)/2/sigma/sigma);
        end
    end
    sum1=sum(o);
    sum2=sum(sum1);
    o=o/sum2;
end

```

