**Practical**

**1)Aim:** 2D Linear Convolution, Circular Convolution between two 2D matrices

Code:

x =[4 ,5 ,6;7 ,8 ,9];

h = [1;1;1];

disp (x, ' x=' )

disp (h, ' h=' )

[y,X,H] = conv2d (x,h);

disp (y, ' Li n e a r 2D c o n v o l u t i o n r e s u l t y =' )

**Li n e a r 2D c o n v o l u t i o n r e s u l t y =**

**4. 5. 6.**

**11. 13. 15.**

**11. 13. 15.**

**7. 8. 9.**

**clc**

x = [1 ,2;3 ,4];

h = [5 ,6;7 ,8];

X = fft2 (x);

H = fft2 (h);

Y = X.\*H;

y = ifft(Y);

disp (y, ' Ci r c u l a r Convo lut i on Re s u l t y =' )

**Ci r c u l a r Convo lut i on Re s u l t y =**

**70. 68.**

**62. 60.**

**2)Aim:** Circular Convolution expressed as linear convolution plus alias

code

clc

x = [1 ,2;3 ,4];

h = [5 ,6;7 ,8];

y = conv2d (x,h);

y1 = [y(: ,1)+y(:,$),y(: ,2) ];

y2 = [y1(: ,1)+y1(:,$),y1(: ,2) ];

disp (y, ' Li n e a r Convo lut i on r e s u l t y=' )

Li n e a r Convo lut i on r e s u l t y=

5. 16. 12.

22. 60. 40.

21. 52. 32.

disp (y1, ' c i r c u l a r c o n v o l u t i o n e x p e s s e d as l i n e a r=' )

c i r c u l a r c o n v o l u t i o n e x p e s s e d as l i n e a r=

17. 16.

62. 60.

53. 52.

disp (y1 , ' circular convoltuion expressed as Li n e a r plus alias=' )

circular convoltuion expressed as Li n e a r plus alias=

17. 16.

62. 60.

53. 52

**3)Aim:** Linear Cross correlation of a 2D matrix, Circular correlation between two signals and Linear auto correlation of a 2D matrix, Linear Cross correlation of a 2D matrix

Code:

**Linear Cross correlation of a 2D matrix**,

clc ;

x = [3 ,1;2 ,4];

h1 = [1 ,5;2 ,3];

h2 = h1 (:,$ : -1:1);

h = h2($: -1:1 ,:);

y = conv2 (x,h)

disp (y, ' Li n e a r c r o s s Co r r e l a t i o n r e s u l t y=' )

Li n e a r c r o s s Co r r e l a t i o n r e s u l t y=

9. 9. 2.

21. 24. 9.

10. 22. 4.

Circular correlation between two signals

clc ;

x = [1 ,5;2 ,4];

h = [3 ,2;4 ,1];

h = h(:,$ : -1:1) ;

h = h($: -1:1 ,:);

X = fft2(x);

H = fft2(h);

Y = X.\*H;

y = ifft(Y);

disp (y, ' Ci r c u l a r Co r r e l a t i o n r e s u l t y=' )

Ci r c u l a r Co r r e l a t i o n r e s u l t y=

37. 23.

35. 25.

Linear auto correlation of a 2D matrix

x1 = [1 ,1;1 ,1];

x2 = x1 (:,$ : -1:1);

x2 = x2($: -1:1 ,:);

x = conv2(x1 ,x2)

x =

1. 2. 1.

2. 4. 2.

1. 2. 1.

l i n e a r c r o s s c o r r e l a t i o n o f a 2D mat r ix

x = [1 ,1;1 ,1];

h1 = [1 ,2;3 ,4];

h2 = h1 (:,$ : -1:1);

h = h2($: -1:1 ,:);

y = conv2(x,h)

y =

4. 7. 3.

6. 10. 4.

2. 3. 1.

4)**Aim:** DFT of 4x4 gray scale image

Code:

a=imread('D:\is.jpg');

i=rgb2gray(a);

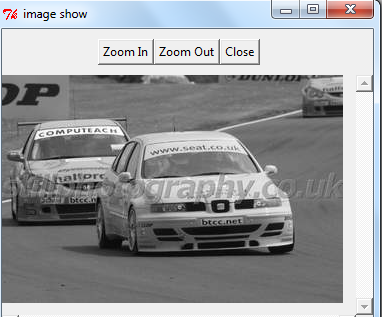
subplot(2,2,1)

imshow(a)



subplot(2,2,2)

imshow(i)



5)**Aim**. Compute discrete cosine transform, Program to perform KL transform for the given 2D matrix

clc ;

N =4; //DCT mat r ix o f o r d e r f o u r

X = dct\_mtx (N);

disp (X, 'DCT mat r ix o f o r d e r f o u r ' )

clear ;

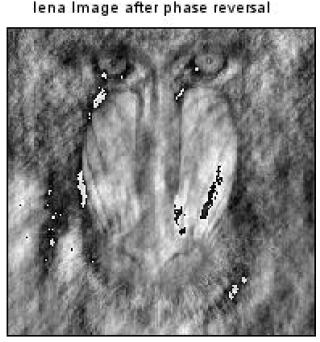
clc ;

X = [4 ,3 ,5 ,6;4 ,2 ,7 ,7;5 ,5 ,6 ,7];

[m,n]= size (X);

A = [];

E = [];



for i =1: n

A = A+X(:,i);

E = E+X(:,i)\*X(:,i) ';

end

mx = A/n;

E = E/n;

C = E - mx\*mx ';

[V,D] = spec (C);

d = diag (D);

[d,i] = gsort (d);

for j = 1: length (d)

T(:,j)= V(:,i(j));

end

T =T

disp (d, ' Eigen Value s a r e U = ' )

disp (T, 'The e i g e n v e c t o r mat r ix T =' )

disp (T, 'The KL t r anf o rm b a s i s i s =' )

//KL t r a n s f o rm

for i = 1:n

Y(:,i)= T\*X(:,i);

end

disp (Y, 'KL t r a n s f o rma t i o n o f the input mat r ix Y =' )

for i = 1:n

x(:,i)= T '\*Y(:,i);

end

disp (x, ' Re c o n s t r u c t mat r ix o f the g i v e n sampl e

mat r ix X =' )

6)**Aim**:Brightness enhancement of an image, Contrast Manipulation, image negative

Brightness enhancement of an image

a=imread('D:\is.jpg');

imshow(a);

for i=1:1 size(image,1)

for j=1:2 size(image,1)

new(i,j)=image(i,j)+50;

end

figure,imshow(new);

for i=1:1 size(image,1

for i=1:1 size(image,2)

new(i,j)=image(i,j)-50;

end

en

end

figure,imshow(new)



Contrast manipulation

a=imread('E:\hsrop.jpg');

imshow(a);

for i=1:1 size(a,1)

for j=1:1 size(a,2)

new(i,j,1)=a(i,j,1)\*2;

new(i,j,1)=a(i,j,1)\*2;

new(i,j,2)=a(i,j,2)\*2;

new(i,j,3)=a(i,j,3)\*2;

end

end

figure,imshow(new);

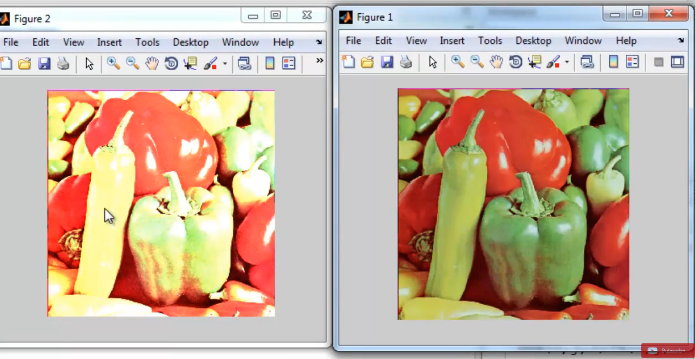


Image Negative

a=imread('D:\is.jpg');

imshow(a);

for i=1:1 size(a,1)

for j=1:1 size(a,2)

b(i,j)=255-a(i,j);

end

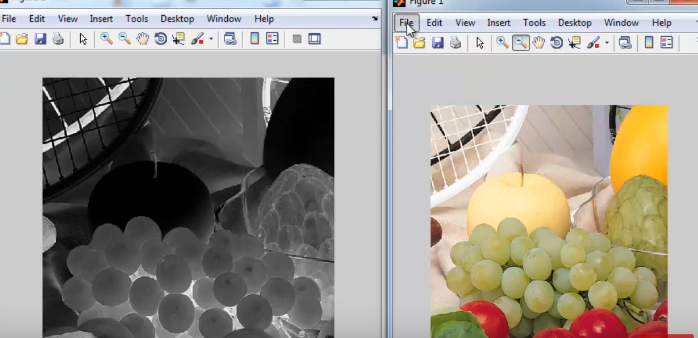
end

ans =

228.

ans =

341.



7. Perform threshold operation, perform gray level slicing without background

clc ;

close ;

a = imread ( 'E: n Di g i t a l Ima g e Pr o c e s s i n g J a y a r ama n n

Chapter5 n l e n a . png ' );

a = rgb2gray (a);

[m n] = size (a);

t = input ( ' Ent e r the t h r e s h o l d parame t e r ' );

for i = 1:m

for j = 1:n

if(a(i,j)<t)

b(i,j)=0;

else

b(i,j) =255;

end

end

end

figure (1)

ShowImage (a, ' Or i g i n a l Image ' );

title ( ' Or i g i n a l Image ' )

figure (2)

ShowImage (b, ' Thr e sho lded Image ' );

title ( ' Thr e sho lded Image ' )

xlabel ( sprintf ( ' Thr e sho ld v a l u e i s %g ' ,t))

// Re s u l t

// Ent e r the t h r e s h o l d parame t e r 140

Program performs gray level slicing without back-

ground



8)Aim: Image Segmentation

Code

clc ;

close

sigma = input ( ' Ent e r the v a l u e o f s igma : ' )

Ent e r the v a l u e o f s igma : 4

sigma =

4.

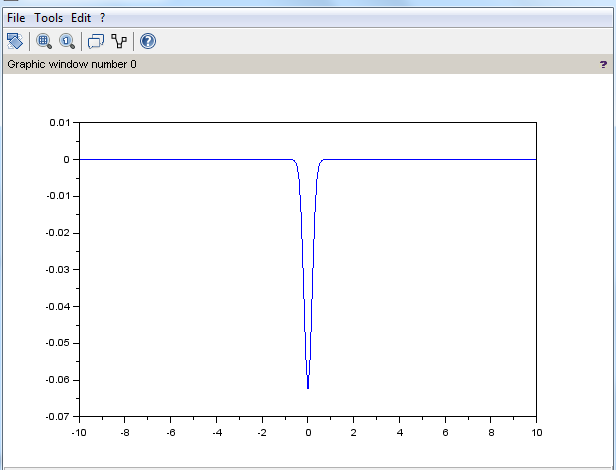
i = -10:.1:10;

j = -10:.1:10;

r= sqrt (i.\*i+j.\*j);

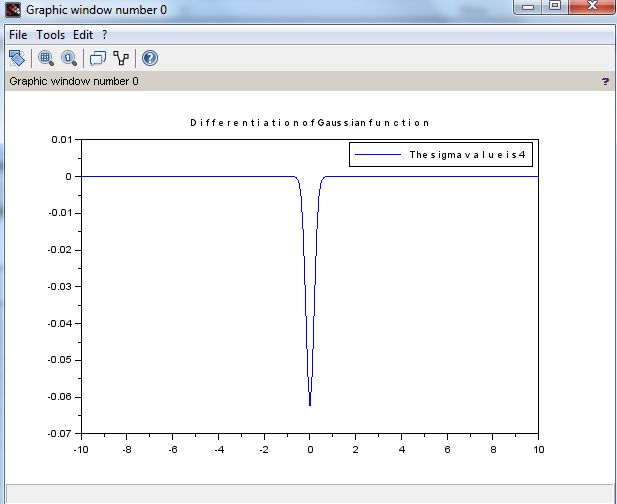
y =(1/( sigma ^2) ) \*((( r.\*r)/ sigma ^2) -1) .\* exp(-r.\*r/2\*sigma ^2) ;

plot (i,y)



legend ( sprintf ( 'The s igma v a l u e i s %g ' ,sigma ))

xtitle ( ' D i f f e r e n t i a t i o n o f Gaus s ian f u n c t i o n ' )



9)Aim Image Compression

Code

Program pe r f o rms Block Trunc a t i on Coding (BTC)

close ;

clear ;

clc ;

x =[65 ,75 ,80 ,70;72 ,75 ,82 ,68;84 ,72 ,62 ,65;66 ,68 ,72 ,80];

disp (x, ' Or i g i n a l Block i s x =' )

Or i g i n a l Block i s x =

65. 75. 80. 70.

72. 75. 82. 68.

84. 72. 62. 65.

66. 68. 72. 80.

[m1 n1 ]= size (x);

blk = input ( ' Ent e r the b l o c k s i z e : ' );

Ent e r the b l o c k s i z e : 4

for i = 1 : blk : m1

for j = 1 : blk : n1

y = x(i:i+( blk -1) ,j:j+( blk -1)) ;

m = mean ( mean (y));

disp (m, 'mean v a l u e i s m =' )

sig = std2 (y);

disp (sig , ' Standard d e v i a t i o n o f the b l o c k i s =' )

b = y > m ;

disp (b, ' Binary a l l o c a t i o n mat r ix i s B=' )

K = sum (sum (b));

disp (K, ' number o f one s =' )

if (K ~= blk ^2 ) & ( K ~= 0)

ml = m- sig \* sqrt (K/(( blk ^2) -K));

disp (ml , 'The v a l u e o f a =' )

mu = m+ sig \* sqrt ((( blk ^2) -K)/K);

disp (mu , 'The v a l u e o f b =' )

x(i:i+( blk -1) , j:j+( blk -1) ) = b\*mu

+(1 - b)\*ml;

end

end

end

mean v a l u e i s m =

72.25

Standard d e v i a t i o n o f the b l o c k i s =

6.6282225

Binary a l l o c a t i o n mat r ix i s B=

F T T F

F T T F

T F F F

F F F T

number o f one s =

6.

The v a l u e o f a =

67.115801

The v a l u e o f b =

80.806998

x =

0. 80.806998 80.806998 0.

0. 80.806998 80.806998 0.

80.806998 0. 0. 0.

0. 0. 0. 80.806998

10)Aim Binary Image Processing and Colour Image processing

Code:

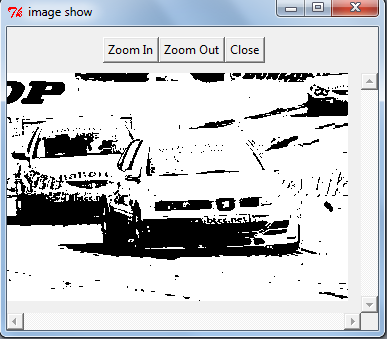
a=imread('D:\is.jpg');

imshow(a)



BW = im2bw(a,0.4);

Imshow(a) figure, imshow(BW)



Colour Image processing

5 clc ;

6 close ;

7 RGB = imread('D:\is.jpg');

//SIVP t o o l b o x

a1 = RGB;

b1 = RGB;

c1 = RGB;

a1 (: ,: ,1) =0;

b1 (: ,: ,2) =0;

c1 (: ,: ,3) =0;

figure (1)

ShowColorImage (RGB , ' Or i g i n a l Co l o r Image ' ); //IPD

t o o l b o x

figure (2)

ShowColorImage (a1 , 'Red Mi s s ing ' );

figure (3)

