**LAB - 7**

**TAMURA FEATURE EXTRACTION**

**PRIYANSHU SHARMA**

**15BCE1282**

**QUESTION - 1**

**Calculate the Tamura features for these images. Tabulate the features as a CSV file.**

**CODE**

clc

clear all

srcFiles = dir('C:\Users\PRIYANSHU SHARMA\Desktop\PRIYANSHU\6 STUDY\MATLAB\LAB 4\\*.jpg');

srcFiles;

for i=1:10

d = strcat('C:\Users\PRIYANSHU SHARMA\Desktop\PRIYANSHU\6 STUDY\MATLAB\LAB 4\',srcFiles(i).name);

IColor = imread(d);

I = rgb2gray(IColor);

[r,c] = size(I);

G=double(I);

%%%---CONTRAST---%%%

[counts,graylevels]=imhist(I);%histogram of image

%%figure,

imhist(I);

title('Gray-level distribution')

PI=counts/(r\*c);

averagevalue=sum(graylevels.\*PI);%mean value

u4=sum((graylevels-repmat(averagevalue,[256,1])).^4.\*PI);%4th moment about mean

variance=sum((graylevels-repmat(averagevalue,[256,1])).^2.\*PI);%variance(2nd moment about mean)

alpha4=u4/variance^2;%kurtosis

%Contrast Value

Fcontrastd=sqrt(variance)/alpha4.^(1/4);

%%%---COARSENESS---%%%

A1=zeros(r,c);A2=zeros(r,c);

A3=zeros(r,c);A4=zeros(r,c);

A5=zeros(r,c);A6=zeros(r,c);

%Sbest for coarseness

Sbest=zeros(r,c);

%Subtracting for Horizontal and Vertical case

E1h=zeros(r,c);E1v=zeros(r,c);

E2h=zeros(r,c);E2v=zeros(r,c);

E3h=zeros(r,c);E3v=zeros(r,c);

E4h=zeros(r,c);E4v=zeros(r,c);

E5h=zeros(r,c);E5v=zeros(r,c);

E6h=zeros(r,c); E6v=zeros(r,c);

flag=0;%To avoid errors

%2x2 E1h and E1v

%subtracting average of neighbouring 2x2 pixels

for x=2:r

for y=2:c

A1(x,y)=(sum(sum(G(x-1:x,y-1:y))));

end

end

for x=2:r-1

for y=2:c-1

E1h(x,y) = A1(x+1,y)-A1(x-1,y);

E1v(x,y) = A1(x,y+1)-A1(x,y-1);

end

end

E1h=E1h/2^(2\*1);

E1v=E1v/2^(2\*1);

%4x4 E2h and E2v

if (r<4||c<4)

flag=1;

end

%subtracting average of neighbouring 4x4 pixels

if(flag==0)

for x=3:r-1

for y=3:c-1

A2(x,y)=(sum(sum(G(x-2:x+1,y-2:y+1))));

end

end

for x=3:r-2

for y=3:c-2

E2h(x,y) = A2(x+2,y)-A2(x-2,y);

E2v(x,y) = A2(x,y+2)-A2(x,y-2);

end

end

end

E2h=E2h/2^(2\*2);

E2v=E2v/2^(2\*2);

%8x8 E3h and E3v

if (r<8||c<8)

flag=1;

end

%subtracting average of neighbouring 8x8 pixels

if(flag==0)

for x=5:r-3

for y=5:c-3

A3(x,y)=(sum(sum(G(x-4:x+3,y-4:y+3))));

end

end

for x=5:r-4

for y=5:c-4

E3h(x,y) = A3(x+4,y)-A3(x-4,y);

E3v(x,y) = A3(x,y+4)-A3(x,y-4);

end

end

end

E3h=E3h/2^(2\*3);

E3v=E3v/2^(2\*3);

%16x16 E4h and E4v

if (r<16||c<16)

flag=1;

end

%subtracting average of neighbouring 16x16 pixels

if(flag==0)

for x=9:r-7

for y=9:c-7

A4(x,y)=(sum(sum(G(x-8:x+7,y-8:y+7))));

end

end

for x=9:r-8

for y=9:c-8

E4h(x,y) = A4(x+8,y)-A4(x-8,y);

E4v(x,y) = A4(x,y+8)-A4(x,y-8);

end

end

end

E4h=E4h/2^(2\*4);

E4v=E4v/2^(2\*4);

%32x32 E5h and E5v

if (r<32||c<32)

flag=1;

end

%subtracting average of neighbouring 32x32 pixels

if(flag==0)

for x=17:r-15

for y=17:c-15

A5(x,y)=(sum(sum(G(x-16:x+15,y-16:y+15))));

end

end

for x=17:r-16

for y=17:c-16

E5h(x,y) = A5(x+16,y)-A5(x-16,y);

E5v(x,y) = A5(x,y+16)-A5(x,y-16);

end

end

end

E5h=E5h/2^(2\*5);

E5v=E5v/2^(2\*5);

%64x64 E6h and E6v

if (r<64||c<64)

flag=1;

end

%subtracting average of neighbouring 64x64 pixels

if(flag==0)

for x=33:r-31

for y=33:c-31

A6(x,y)=(sum(sum(G(x-32:x+31,y-32:y+31))));

end

end

for x=33:r-32

for y=33:c-32

E6h(x,y) = A6(x+32,y)-A6(x-32,y);

E6v(x,y) = A6(x,y+32)-A6(x,y-32);

end

end

end

E6h=E6h/2^(2\*6);

E6v=E6v/2^(2\*6);

%plots

%%figure

%%subplot(131);

%%imshow(IColor);

title('Original image')

%%subplot(132);

%%imshow(E1h);

title('Horizontal case')

%%subplot(133)

%%imshow(E1v);

title('Vertical case')

%at each point pick best size "Sbest", which gives highest output value

for i=1:r

for j=1:c

[maxv,index]=max([abs(E1h(i,j)),abs(E1v(i,j)),abs(E2h(i,j)),abs(E2v(i,j)),...

abs(E3h(i,j)),abs(E3v(i,j)),abs(E4h(i,j)),abs(E4v(i,j)),abs(E5h(i,j)),...

abs(E5v(i,j)),abs(E6h(i,j)),abs(E6v(i,j))]);

k=floor((index+1)/2);%'k'corresponding to highest E in either direction

Sbest(i,j)=2.^k;

end

end

%%figure;

plot(Sbest)

title('Output of best size detector')

%Coarseness Value

Fcoarsenessd=sum(sum(Sbest))/(r\*c);

%%%---DIRECTION---%%%

PrewittH = [-1 0 1;-1 0 1;-1 0 1];%for measuring horizontal differences

PrewittV = [1 1 1;0 0 0;-1 -1 -1];%for measuring vertical differences

%Applying PerwittH operator

deltaH=zeros(r,c);

for i=2:r-1

for j=2:c-1

deltaH(i,j)=sum(sum(G(i-1:i+1,j-1:j+1).\*PrewittH));

end

end

%Modifying borders

for j=2:c-1

deltaH(1,j)=G(1,j+1)-G(1,j);

deltaH(r,j)=G(r,j+1)-G(r,j);

end

for i=1:r

deltaH(i,1)=G(i,2)-G(i,1);

deltaH(i,c)=G(i,c)-G(i,c-1);

end

%Applying PerwittV operator

deltaV=zeros(r,c);

for i=2:r-1

for j=2:c-1

deltaV(i,j)=sum(sum(G(i-1:i+1,j-1:j+1).\*PrewittV));

end

end

%Modifying borders

for j=1:c

deltaV(1,j)=G(2,j)-G(1,j);

deltaV(r,j)=G(r,j)-G(r-1,j);

end

for i=2:r-1

deltaV(i,1)=G(i+1,1)-G(i,1);

deltaV(i,c)=G(i+1,c)-G(i,c);

end

%Magnitude

deltaG=(abs(deltaH)+abs(deltaV))/2;

%Local edge direction (0<=theta<pi)

theta=zeros(r,c);

for i=1:r

for j=1:c

if (deltaH(i,j)==0)&&(deltaV(i,j)==0)

theta(i,j)=0;

elseif deltaH(i,j)==0

theta(i,j)=pi;

else

theta(i,j)=atan(deltaV(i,j)/deltaH(i,j))+pi/2;

end

end

end

deltaGt = deltaG(:);

theta1=theta(:);

%Set a Threshold value for delta G

n = 16;

HD = zeros(1,n);

Threshold=12;

counti=0;

for m=0:(n-1)

countk=0;

for k = 1:length(deltaGt)

if ((deltaGt(k)>=Threshold) && (theta1(k)>=(2\*m-1)\*pi/(2\*n)) && (theta1(k)<(2\*m+1)\*pi/(2\*n)))

countk=countk+1;

counti=counti+1;

end

end

HD(m+1) = countk;

end

HDf = HD/counti;

%%figure;

plot(HDf);

title('Local Directionality Histogram HDf')

%findpeaks function to find peak values

[m p]=findpeaks(HDf,0.000005);

Fd=0;

for np = 1:length(m)

phaiP=m(np)\*(pi/n);

for phi=1:length(HDf)

Fd=Fd+(phi\*(pi/n)-phaiP)^2\*HDf(phi);

end

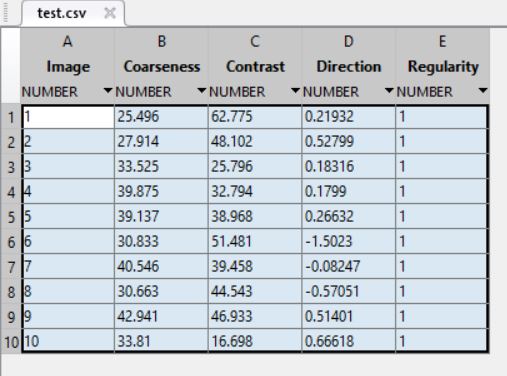
end

r = (1/n);

Fdirectiond = 1 - r\*np\*Fd;

end

**RESULT**

****

**QUESTION - 2**

**Implement a CBIR system that uses Tamura features as image descriptors.**

**CODE**

clc

clear all

q=imread('C:\Users\PRIYANSHU SHARMA\Desktop\PRIYANSHU\6 STUDY\MATLAB\LAB 4\1.jpg');

%--CONTRAST--%

I = rgb2gray(q);

[r,c] = size(I);

G=double(I);

[counts,graylevels]=imhist(I);%histogram of image

%figure,

imhist(I);

title('Gray-level distribution')

PI=counts/(r\*c);

averagevalue=sum(graylevels.\*PI);%mean value

u4=sum((graylevels-repmat(averagevalue,[256,1])).^4.\*PI);%4th moment about mean

variance=sum((graylevels-repmat(averagevalue,[256,1])).^2.\*PI);%variance(2nd moment about mean)

alpha4=u4/variance^2;%kurtosis

%Contrast Value

Fcontrast=sqrt(variance)/alpha4.^(1/4);

%--COARSENESS--%

I = rgb2gray(q);

[r,c] = size(I);

G=double(I);

A1=zeros(r,c);A2=zeros(r,c);

A3=zeros(r,c);A4=zeros(r,c);

A5=zeros(r,c);A6=zeros(r,c);

%Sbest for coarseness

Sbest=zeros(r,c);

%Subtracting for Horizontal and Vertical case

E1h=zeros(r,c);E1v=zeros(r,c);

E2h=zeros(r,c);E2v=zeros(r,c);

E3h=zeros(r,c);E3v=zeros(r,c);

E4h=zeros(r,c);E4v=zeros(r,c);

E5h=zeros(r,c);E5v=zeros(r,c);

E6h=zeros(r,c); E6v=zeros(r,c);

flag=0;%To avoid errors

%2x2 E1h and E1v

%subtracting average of neighbouring 2x2 pixels

for x=2:r

for y=2:c

A1(x,y)=(sum(sum(G(x-1:x,y-1:y))));

end

end

for x=2:r-1

for y=2:c-1

E1h(x,y) = A1(x+1,y)-A1(x-1,y);

E1v(x,y) = A1(x,y+1)-A1(x,y-1);

end

end

E1h=E1h/2^(2\*1);

E1v=E1v/2^(2\*1);

%4x4 E2h and E2v

if (r<4||c<4)

flag=1;

end

%subtracting average of neighbouring 4x4 pixels

if(flag==0)

for x=3:r-1

for y=3:c-1

A2(x,y)=(sum(sum(G(x-2:x+1,y-2:y+1))));

end

end

for x=3:r-2

for y=3:c-2

E2h(x,y) = A2(x+2,y)-A2(x-2,y);

E2v(x,y) = A2(x,y+2)-A2(x,y-2);

end

end

end

E2h=E2h/2^(2\*2);

E2v=E2v/2^(2\*2);

%8x8 E3h and E3v

if (r<8||c<8)

flag=1;

end

%subtracting average of neighbouring 8x8 pixels

if(flag==0)

for x=5:r-3

for y=5:c-3

A3(x,y)=(sum(sum(G(x-4:x+3,y-4:y+3))));

end

end

for x=5:r-4

for y=5:c-4

E3h(x,y) = A3(x+4,y)-A3(x-4,y);

E3v(x,y) = A3(x,y+4)-A3(x,y-4);

end

end

end

E3h=E3h/2^(2\*3);

E3v=E3v/2^(2\*3);

%16x16 E4h and E4v

if (r<16||c<16)

flag=1;

end

%subtracting average of neighbouring 16x16 pixels

if(flag==0)

for x=9:r-7

for y=9:c-7

A4(x,y)=(sum(sum(G(x-8:x+7,y-8:y+7))));

end

end

for x=9:r-8

for y=9:c-8

E4h(x,y) = A4(x+8,y)-A4(x-8,y);

E4v(x,y) = A4(x,y+8)-A4(x,y-8);

end

end

end

E4h=E4h/2^(2\*4);

E4v=E4v/2^(2\*4);

%32x32 E5h and E5v

if (r<32||c<32)

flag=1;

end

%subtracting average of neighbouring 32x32 pixels

if(flag==0)

for x=17:r-15

for y=17:c-15

A5(x,y)=(sum(sum(G(x-16:x+15,y-16:y+15))));

end

end

for x=17:r-16

for y=17:c-16

E5h(x,y) = A5(x+16,y)-A5(x-16,y);

E5v(x,y) = A5(x,y+16)-A5(x,y-16);

end

end

end

E5h=E5h/2^(2\*5);

E5v=E5v/2^(2\*5);

%64x64 E6h and E6v

if (r<64||c<64)

flag=1;

end

%subtracting average of neighbouring 64x64 pixels

if(flag==0)

for x=33:r-31

for y=33:c-31

A6(x,y)=(sum(sum(G(x-32:x+31,y-32:y+31))));

end

end

for x=33:r-32

for y=33:c-32

E6h(x,y) = A6(x+32,y)-A6(x-32,y);

E6v(x,y) = A6(x,y+32)-A6(x,y-32);

end

end

end

E6h=E6h/2^(2\*6);

E6v=E6v/2^(2\*6);

%plots

%figure

%subplot(131);

%imshow(IColor);

title('Original image')

%subplot(132);

%imshow(E1h);

title('Horizontal case')

%subplot(133)

%imshow(E1v);

title('Vertical case')

%at each point pick best size "Sbest", which gives highest output value

for i=1:r

for j=1:c

[maxv,index]=max([abs(E1h(i,j)),abs(E1v(i,j)),abs(E2h(i,j)),abs(E2v(i,j)),...

abs(E3h(i,j)),abs(E3v(i,j)),abs(E4h(i,j)),abs(E4v(i,j)),abs(E5h(i,j)),...

abs(E5v(i,j)),abs(E6h(i,j)),abs(E6v(i,j))]);

k=floor((index+1)/2);%'k'corresponding to highest E in either direction

Sbest(i,j)=2.^k;

end

end

figure;

plot(Sbest)

title('Output of best size detector')

%Coarseness Value

Fcoarseness=sum(sum(Sbest))/(r\*c);

%--DIRECTION--%

I = rgb2gray(q);

[r,c] = size(I);

G=double(I);

PrewittH = [-1 0 1;-1 0 1;-1 0 1];%for measuring horizontal differences

PrewittV = [1 1 1;0 0 0;-1 -1 -1];%for measuring vertical differences

%Applying PerwittH operator

deltaH=zeros(r,c);

for i=2:r-1

for j=2:c-1

deltaH(i,j)=sum(sum(G(i-1:i+1,j-1:j+1).\*PrewittH));

end

end

%Modifying borders

for j=2:c-1

deltaH(1,j)=G(1,j+1)-G(1,j);

deltaH(r,j)=G(r,j+1)-G(r,j);

end

for i=1:r

deltaH(i,1)=G(i,2)-G(i,1);

deltaH(i,c)=G(i,c)-G(i,c-1);

end

%Applying PerwittV operator

deltaV=zeros(r,c);

for i=2:r-1

for j=2:c-1

deltaV(i,j)=sum(sum(G(i-1:i+1,j-1:j+1).\*PrewittV));

end

end

%Modifying borders

for j=1:c

deltaV(1,j)=G(2,j)-G(1,j);

deltaV(r,j)=G(r,j)-G(r-1,j);

end

for i=2:r-1

deltaV(i,1)=G(i+1,1)-G(i,1);

deltaV(i,c)=G(i+1,c)-G(i,c);

end

%Magnitude

deltaG=(abs(deltaH)+abs(deltaV))/2;

%Local edge direction (0<=theta<pi)

theta=zeros(r,c);

for i=1:r

for j=1:c

if (deltaH(i,j)==0)&&(deltaV(i,j)==0)

theta(i,j)=0;

elseif deltaH(i,j)==0

theta(i,j)=pi;

else

theta(i,j)=atan(deltaV(i,j)/deltaH(i,j))+pi/2;

end

end

end

deltaGt = deltaG(:);

theta1=theta(:);

%Set a Threshold value for delta G

n = 16;

HD = zeros(1,n);

Threshold=12;

counti=0;

for m=0:(n-1)

countk=0;

for k = 1:length(deltaGt)

if ((deltaGt(k)>=Threshold) && (theta1(k)>=(2\*m-1)\*pi/(2\*n)) && (theta1(k)<(2\*m+1)\*pi/(2\*n)))

countk=countk+1;

counti=counti+1;

end

end

HD(m+1) = countk;

end

HDf = HD/counti;

%figure;

plot(HDf);

title('Local Directionality Histogram HDf')

%peakdet function to find peak values

[m p]=findpeaks(HDf,0.000005);

Fd=0;

for np = 1:length(m)

phaiP=m(np)\*(pi/n);

for phi=1:length(HDf)

Fd=Fd+(phi\*(pi/n)-phaiP)^2\*HDf(phi);

end

end

r = (1/n);

Fdirection = 1 - r\*np\*Fd;

%--DATABASE--%

fr=[];

srcFiles = dir('C:\Users\PRIYANSHU SHARMA\Desktop\PRIYANSHU\6 STUDY\MATLAB\LAB 4\\*.jpg');

srcFiles;

for i=2:10

d = strcat('C:\Users\PRIYANSHU SHARMA\Desktop\PRIYANSHU\6 STUDY\MATLAB\LAB 4\',srcFiles(i).name);

IColor = imread(d);

I = rgb2gray(IColor);

[r,c] = size(I);

G=double(I);

%%%---CONTRAST---%%%

[counts,graylevels]=imhist(I);%histogram of image

%%figure,

imhist(I);

title('Gray-level distribution')

PI=counts/(r\*c);

averagevalue=sum(graylevels.\*PI);%mean value

u4=sum((graylevels-repmat(averagevalue,[256,1])).^4.\*PI);%4th moment about mean

variance=sum((graylevels-repmat(averagevalue,[256,1])).^2.\*PI);%variance(2nd moment about mean)

alpha4=u4/variance^2;%kurtosis

%Contrast Value

Fcontrastd=sqrt(variance)/alpha4.^(1/4);

%%%---COARSENESS---%%%

A1=zeros(r,c);A2=zeros(r,c);

A3=zeros(r,c);A4=zeros(r,c);

A5=zeros(r,c);A6=zeros(r,c);

%Sbest for coarseness

Sbest=zeros(r,c);

%Subtracting for Horizontal and Vertical case

E1h=zeros(r,c);E1v=zeros(r,c);

E2h=zeros(r,c);E2v=zeros(r,c);

E3h=zeros(r,c);E3v=zeros(r,c);

E4h=zeros(r,c);E4v=zeros(r,c);

E5h=zeros(r,c);E5v=zeros(r,c);

E6h=zeros(r,c); E6v=zeros(r,c);

flag=0;%To avoid errors

%2x2 E1h and E1v

%subtracting average of neighbouring 2x2 pixels

for x=2:r

for y=2:c

A1(x,y)=(sum(sum(G(x-1:x,y-1:y))));

end

end

for x=2:r-1

for y=2:c-1

E1h(x,y) = A1(x+1,y)-A1(x-1,y);

E1v(x,y) = A1(x,y+1)-A1(x,y-1);

end

end

E1h=E1h/2^(2\*1);

E1v=E1v/2^(2\*1);

%4x4 E2h and E2v

if (r<4||c<4)

flag=1;

end

%subtracting average of neighbouring 4x4 pixels

if(flag==0)

for x=3:r-1

for y=3:c-1

A2(x,y)=(sum(sum(G(x-2:x+1,y-2:y+1))));

end

end

for x=3:r-2

for y=3:c-2

E2h(x,y) = A2(x+2,y)-A2(x-2,y);

E2v(x,y) = A2(x,y+2)-A2(x,y-2);

end

end

end

E2h=E2h/2^(2\*2);

E2v=E2v/2^(2\*2);

%8x8 E3h and E3v

if (r<8||c<8)

flag=1;

end

%subtracting average of neighbouring 8x8 pixels

if(flag==0)

for x=5:r-3

for y=5:c-3

A3(x,y)=(sum(sum(G(x-4:x+3,y-4:y+3))));

end

end

for x=5:r-4

for y=5:c-4

E3h(x,y) = A3(x+4,y)-A3(x-4,y);

E3v(x,y) = A3(x,y+4)-A3(x,y-4);

end

end

end

E3h=E3h/2^(2\*3);

E3v=E3v/2^(2\*3);

%16x16 E4h and E4v

if (r<16||c<16)

flag=1;

end

%subtracting average of neighbouring 16x16 pixels

if(flag==0)

for x=9:r-7

for y=9:c-7

A4(x,y)=(sum(sum(G(x-8:x+7,y-8:y+7))));

end

end

for x=9:r-8

for y=9:c-8

E4h(x,y) = A4(x+8,y)-A4(x-8,y);

E4v(x,y) = A4(x,y+8)-A4(x,y-8);

end

end

end

E4h=E4h/2^(2\*4);

E4v=E4v/2^(2\*4);

%32x32 E5h and E5v

if (r<32||c<32)

flag=1;

end

%subtracting average of neighbouring 32x32 pixels

if(flag==0)

for x=17:r-15

for y=17:c-15

A5(x,y)=(sum(sum(G(x-16:x+15,y-16:y+15))));

end

end

for x=17:r-16

for y=17:c-16

E5h(x,y) = A5(x+16,y)-A5(x-16,y);

E5v(x,y) = A5(x,y+16)-A5(x,y-16);

end

end

end

E5h=E5h/2^(2\*5);

E5v=E5v/2^(2\*5);

%64x64 E6h and E6v

if (r<64||c<64)

flag=1;

end

%subtracting average of neighbouring 64x64 pixels

if(flag==0)

for x=33:r-31

for y=33:c-31

A6(x,y)=(sum(sum(G(x-32:x+31,y-32:y+31))));

end

end

for x=33:r-32

for y=33:c-32

E6h(x,y) = A6(x+32,y)-A6(x-32,y);

E6v(x,y) = A6(x,y+32)-A6(x,y-32);

end

end

end

E6h=E6h/2^(2\*6);

E6v=E6v/2^(2\*6);

%plots

%%figure

%%subplot(131);

%%imshow(IColor);

title('Original image')

%%subplot(132);

%%imshow(E1h);

title('Horizontal case')

%%subplot(133)

%%imshow(E1v);

title('Vertical case')

%at each point pick best size "Sbest", which gives highest output value

for i=1:r

for j=1:c

[maxv,index]=max([abs(E1h(i,j)),abs(E1v(i,j)),abs(E2h(i,j)),abs(E2v(i,j)),...

abs(E3h(i,j)),abs(E3v(i,j)),abs(E4h(i,j)),abs(E4v(i,j)),abs(E5h(i,j)),...

abs(E5v(i,j)),abs(E6h(i,j)),abs(E6v(i,j))]);

k=floor((index+1)/2);%'k'corresponding to highest E in either direction

Sbest(i,j)=2.^k;

end

end

%%figure;

plot(Sbest)

title('Output of best size detector')

%Coarseness Value

Fcoarsenessd=sum(sum(Sbest))/(r\*c);

%%%---DIRECTION---%%%

PrewittH = [-1 0 1;-1 0 1;-1 0 1];%for measuring horizontal differences

PrewittV = [1 1 1;0 0 0;-1 -1 -1];%for measuring vertical differences

%Applying PerwittH operator

deltaH=zeros(r,c);

for i=2:r-1

for j=2:c-1

deltaH(i,j)=sum(sum(G(i-1:i+1,j-1:j+1).\*PrewittH));

end

end

%Modifying borders

for j=2:c-1

deltaH(1,j)=G(1,j+1)-G(1,j);

deltaH(r,j)=G(r,j+1)-G(r,j);

end

for i=1:r

deltaH(i,1)=G(i,2)-G(i,1);

deltaH(i,c)=G(i,c)-G(i,c-1);

end

%Applying PerwittV operator

deltaV=zeros(r,c);

for i=2:r-1

for j=2:c-1

deltaV(i,j)=sum(sum(G(i-1:i+1,j-1:j+1).\*PrewittV));

end

end

%Modifying borders

for j=1:c

deltaV(1,j)=G(2,j)-G(1,j);

deltaV(r,j)=G(r,j)-G(r-1,j);

end

for i=2:r-1

deltaV(i,1)=G(i+1,1)-G(i,1);

deltaV(i,c)=G(i+1,c)-G(i,c);

end

%Magnitude

deltaG=(abs(deltaH)+abs(deltaV))/2;

%Local edge direction (0<=theta<pi)

theta=zeros(r,c);

for i=1:r

for j=1:c

if (deltaH(i,j)==0)&&(deltaV(i,j)==0)

theta(i,j)=0;

elseif deltaH(i,j)==0

theta(i,j)=pi;

else

theta(i,j)=atan(deltaV(i,j)/deltaH(i,j))+pi/2;

end

end

end

deltaGt = deltaG(:);

theta1=theta(:);

%Set a Threshold value for delta G

n = 16;

HD = zeros(1,n);

Threshold=12;

counti=0;

for m=0:(n-1)

countk=0;

for k = 1:length(deltaGt)

if ((deltaGt(k)>=Threshold) && (theta1(k)>=(2\*m-1)\*pi/(2\*n)) && (theta1(k)<(2\*m+1)\*pi/(2\*n)))

countk=countk+1;

counti=counti+1;

end

end

HD(m+1) = countk;

end

HDf = HD/counti;

%%figure;

plot(HDf);

title('Local Directionality Histogram HDf')

%findpeaks function to find peak values

[m p]=findpeaks(HDf,0.000005);

Fd=0;

for np = 1:length(m)

phaiP=m(np)\*(pi/n);

for phi=1:length(HDf)

Fd=Fd+(phi\*(pi/n)-phaiP)^2\*HDf(phi);

end

end

r = (1/n);

Fdirectiond = 1 - r\*np\*Fd;

fr(i)=(abs(Fcontrastd-Fcontrast)+abs(Fcoarsenessd-Fcoarseness)+(Fdirectiond-Fdirection))/3

end

clear min;

ra = [];

fr(1)=[];

fr;

ascen=sort(fr);

ascen;

[m,n] = size(fr);

for k=1:n

mini=min(fr);

ra(k)=find(fr==ascen(k))+1;

fr;

end

ra;

for i=1:9

name = strcat(num2str(ra(i)),'.jpg');

filename = strcat('C:\Users\PRIYANSHU SHARMA\Desktop\PRIYANSHU\6 STUDY\MATLAB\LAB 4\',name);

result = imread(filename);

figure;

image(result);

end

fr

ascen

ra

**KLDiv.m**

function dist=KLDiv(P,Q)

% dist = KLDiv(P,Q) Kullback-Leibler divergence of two discrete probability

% distributions

% P and Q are automatically normalised to have the sum of one on rows

% have the length of one at each

% P = n x nbins

% Q = 1 x nbins or n x nbins(one to one)

% dist = n x 1

if size(P,2)~=size(Q,2)

error('the number of columns in P and Q should be the same');

end

if sum(~isfinite(P(:))) + sum(~isfinite(Q(:)))

error('the inputs contain non-finite values!')

end

% normalizing the P and Q

if size(Q,1)==1

Q = Q ./sum(Q);

P = P ./repmat(sum(P,2),[1 size(P,2)]);

temp = P.\*log(P./repmat(Q,[size(P,1) 1]));

temp(isnan(temp))=0;% resolving the case when P(i)==0

dist = sum(temp,2);

elseif size(Q,1)==size(P,1)

Q = Q ./repmat(sum(Q,2),[1 size(Q,2)]);

P = P ./repmat(sum(P,2),[1 size(P,2)]);

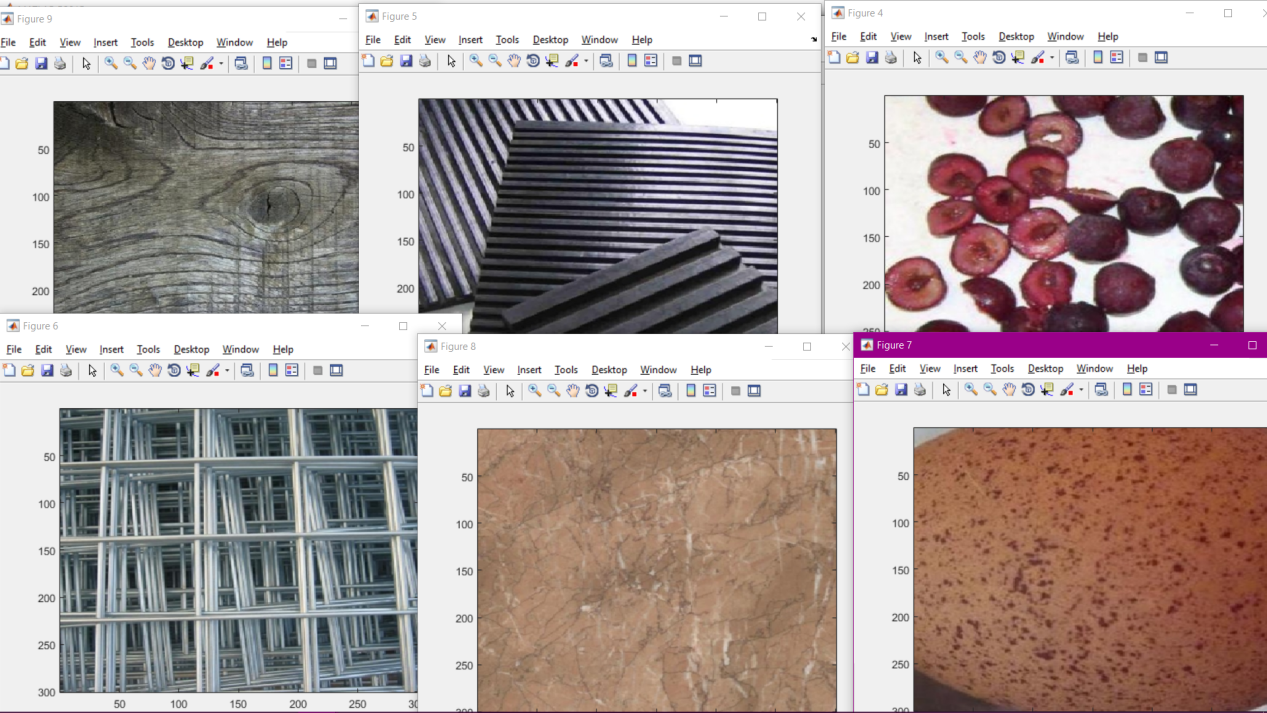
temp = P.\*log(P./Q);

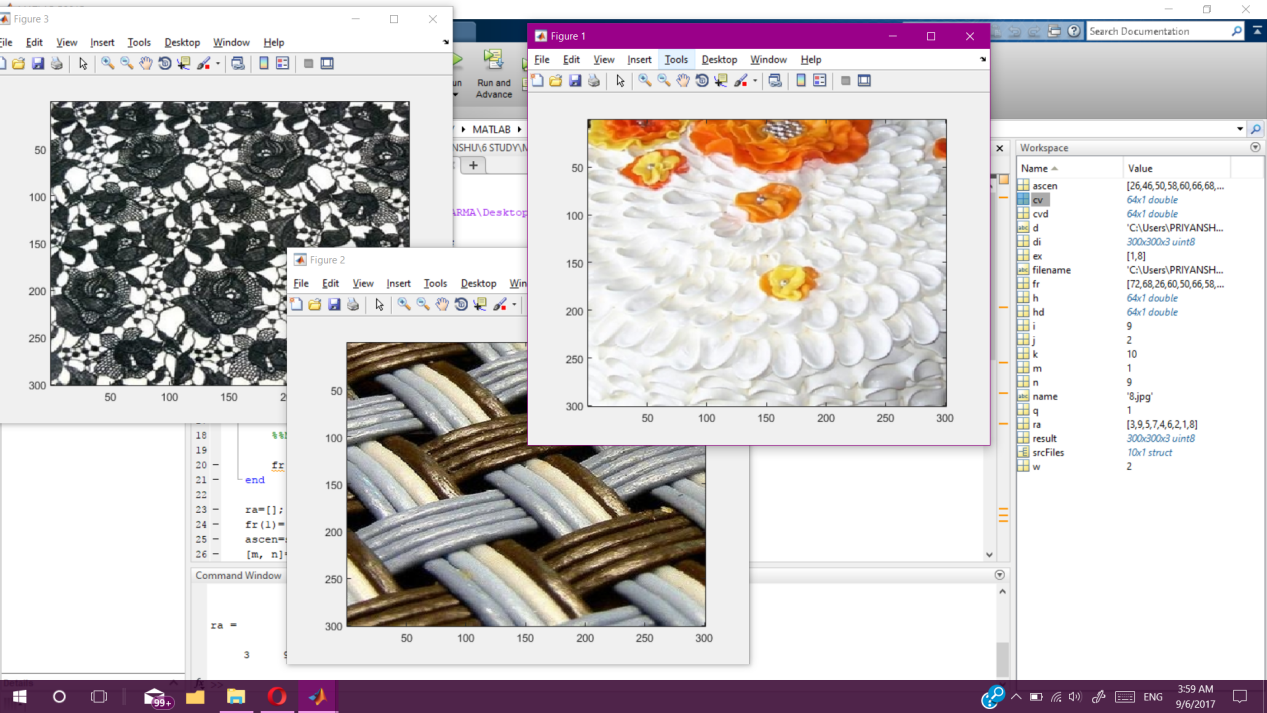
temp(isnan(temp))=0; % resolving the case when P(i)==0

dist = sum(temp,2);

end

**RESULT**

****

****