**ACCESSING IMAGES FROM THE FOLDER CLASS -1 AND STORING IT IN A STRUCTURE**

mydir='/MATLAB Drive/Published/Birds-1/Class-1';

fileformat='\*.jpg';

dd=dir(fullfile(mydir,fileformat));

assert(numel(dd) > 0, 'No file was found. Check that the path is correct');

my\_img = struct('img', cell(size(dd)));

for zz=1:numel(dd)

my\_img(zz).img = imread(fullfile(mydir,dd(zz).name));

end

**CREATING A STRUCTURE FOR R COMPONENT**

r\_img = struct('img', cell(size(dd)));

for zz=1:numel(dd)

r\_img(zz).img = my\_img(zz).img(:,:,1);

end

**CREATING A STRUCTURE FOR G COMPONENT**

g\_img = struct('img', cell(size(dd)));

for zz=1:numel(dd)

g\_img(zz).img = my\_img(zz).img(:,:,2);

end

**CREATING A STRUCTURE FOR B COMPONENT**

b\_img = struct('img', cell(size(dd)));

for zz=1:numel(dd)

r\_img(zz).img = my\_img(zz).img(:,:,3);

end

**CREATING A STRUCTURE FOR GRAY SCALE VERSION**

gray\_img = struct('img', cell(size(dd)));

for zz=1:numel(dd)

gray\_img(zz).img = rgb2gray(my\_img(zz).img);

end

**CREATING A STRUCTURE FOR SHARPENED**

edge\_img = struct('img', cell(size(dd)));

for zz=1:numel(dd)

edge\_img(zz).img = fourrier(gray\_img(zz).img,0.09,4);

end

**FUNCTION TO SHARPEN THE IMAGE**

function outimg = fourrier(inimg,f,ch)

q=3;

[m,n]=size(inimg);

fft=fftshift(fft2(inimg));

f=f\*sqrt(m^2+n^2);

h=zeros(m,n);

for i=1:m

for j=1:n

d=sqrt(abs(i-m/2)^2+abs(j-n/2)^2);

if ch==1 %lf-ideal

if(d < f)

h(i,j)=1;

end

elseif ch==2 %lf-butter

h(i,j)=1/(1+(d/f)^(2\*n));

elseif ch==3 %lf-gaussian

h(i,j)=exp(-(d^2)/(2\*f^2));

elseif ch==4 %hf-ideal

if(d > f)

h(i,j)=1;

end

elseif ch==5 %hf-butter

h(i,j)=1/(1+(f/d)^(2\*n));

elseif ch==6 %hf-gaussian

h(i,j)=1-exp(-(d^2)/(2\*f^2));

else

break

end

end

end

g=h.\*fft;

g=ifftshift(g);

outimg=uint8(real(ifft2(g)));

end

**CODE TO EXTRACT FEACTURES IN THE FREQUENCY DOMAIN AFTER APPLYING FFT**

%fft

%gray

for i=1:numel(dd)

current=gray\_img(i).img; %r\_img(i).img

%fourier transform

fft\_img=fft2(current);

%statistical measures

av=real(mean(mean(fft\_img)));

med=real(median(median(fft\_img)));

st\_dev=real(std(std(double(fft\_img))));

max\_=real(max(max(fft\_img)));

min\_=real(min(min(fft\_img)));

%Column Values

rgb=[av,med,st\_dev,max\_,min\_];

writematrix(rgb,'IVA\_FFT\_gray.csv','WriteMode', 'append');

end

**CODE TO EXTRACT FEACTURES IN THE FREQUENCY DOMAIN AFTER APPLYING DCT**

%dct

%gray

for i=1:numel(dd)

current=gray\_img(i).img; %r\_img(i).img

%dct

dct\_img=dct2(current);

dc=dct\_img(1,1);

writematrix(dc,'IVA\_DCT\_gray.csv','WriteMode', 'append')

end

**CODE TO EXTRACT FEACTURES IN THE FREQUENCY DOMAIN AFTER APPLYING WAVELET TRANSFORM**

%wavelet

%gray

for i=1:numel(dd)

current=gray\_img(i).img; %r\_img(i).img

%WAVELET transform

wave\_img=wave(current,'haar',3);

%statistical measures

av=real(mean(mean(wave\_img)));

med=real(median(median(wave\_img)));

st\_dev=real(std(std(double(wave\_img))));

max\_=real(max(max(wave\_img)));

min\_=real(min(min(wave\_img)));

%Column Values

rgb=[av,med,st\_dev,max\_,min\_];

writematrix(rgb,'IVA\_wavelet\_gray.csv','WriteMode', 'append');

end

**FUNCTION GET THE WAVELET TRANSFORM**

function A3 = wave(im, wname, no\_levels)

[C, S]=wavedec2(im2double(im),no\_levels,wname);

A3=appcoef2(C,S,wname,no\_levels);

end