Hybrid image

**HPF image:**

**LPF Image:**



**Hybrid Image Source Codes:**

% load in my and friend's images

myImageDir='me3.png';

friendImageDir='friend.png';

% read it in and gray scale it

myImage=im2double( rgb2gray( imread(myImageDir) ) );

friendImage=im2double( rgb2gray( imread(friendImageDir) ) );

% setting gaussian matrix multiplication

sigma=6;

gaussianDim=3\*sigma\*2+1;

myImageHPFed=HPF(myImage,gaussianDim);

figure('Name','high pass filtered my image'), imshow(myImageHPFed,[]);

sigma=1;

gaussianDim=3\*sigma\*2+1;

friendImageLPF=LPF(friendImage,gaussianDim);

figure('Name','low pass filtered friend image'), imshow(friendImageLPF,[]);

hybridedImage=friendImageLPF+myImageHPFed;

figure('Name','Hybrided Image'), imshow(hybridedImage, []);

**high pass filter function source code:**

function HPF\_Image=HPF(image,filter\_size)

% fouier and shift the image

fourierImage = fft2(image);

fourierShiftedImage=fftshift(fourierImage);

% alloting size of image

[i ,j]=size(fourierImage);

% looping through the iteration

X=0:j-1;

Y=0:i-1;

% mesh grid

[X, Y]=meshgrid(X,Y);

Center\_of\_x=0.5\*j;

Center\_of\_y=0.5\*i;

% h(t) for fourier transformation

G=1-exp(-((X-Center\_of\_x).^2+(Y-Center\_of\_y).^2)./(2\*filter\_size).^2);

% fourer transform and shift for High Pass Filtered Images

fourierFilterImage=fourierShiftedImage.\*G;

fourierFilteredShiftedImage=ifftshift(fourierFilterImage);

HPF\_Image=ifft2(fourierFilteredShiftedImage);

end

**LPF function source codes:**

% function for images undergoing low-pass filter

function LPF\_Image=LPF(image,filter\_size)

% fourier and shifted

fourierImage = fft2(image);

fourierShiftedImage=fftshift(fourierImage);

% size of the image

[i, j]=size(fourierImage);

% iterating through matrix to add values

X=0:j-1;

Y=0:i-1;

% mesh grid

[X, Y]=meshgrid(X,Y);

Center\_of\_x=0.5\*j;

Center\_of\_y=0.5\*i;

% h(x) to be used

G=exp(-((X-Center\_of\_x).^2+(Y-Center\_of\_y).^2)./(2\*filter\_size).^2);

% fourier transform and shift for low pass filter process

image\_FFT\_filtered=fourierShiftedImage.\*G;

image\_FFT\_filtered\_shifted=ifftshift(image\_FFT\_filtered);

LPF\_Image=ifft2(image\_FFT\_filtered\_shifted);

end