Assignment 2

201501146

1) gauss

```
function output = gaussian(im,level)
    fs = 10;
    sig = 100;

    gauss = zeros(fs,fs);
    for i = 1:fs
        for j = 1:fs
            gauss(i,j) =  exp(-(abs(fs - i)^2 + abs(fs - j)^2) / (2*sig*sig));
        end
    end

    N = sum(gauss(:));
    gauss = gauss / N;
    hp = cell(1,level);
    imArr = cell(1,level);
    imFinal = im;

    for i = 1:level
        imTemp = imFinal;

        imArr{i} = imFinal;
        imFinal = imfilter(imFinal, gauss);
        hp{i} = imTemp - imFinal;
        imFinal = imFinal(1:2:end, 1:2:end,:);
    end
    output = {imArr,hp};
end
```

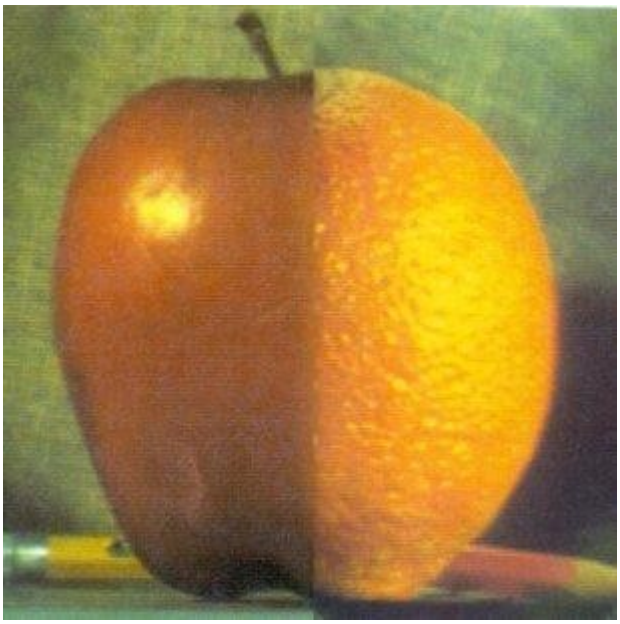
1) laplacian

```
function l = laplacian(im,level)
    imArr = gaussian(im,level);
    hp = cell(1,level);
    imArr = imArr{1};

    for i = level:-1:2
        y = imArr{i - 1};
        x = imArr{i};
        imFinal = imresize(x,[size(y,1) size(y,2)]);
        hp{i - 1} = y - imFinal;
    end
    hp{level} = imArr{level};

    l = hp;
end
```

1) blending

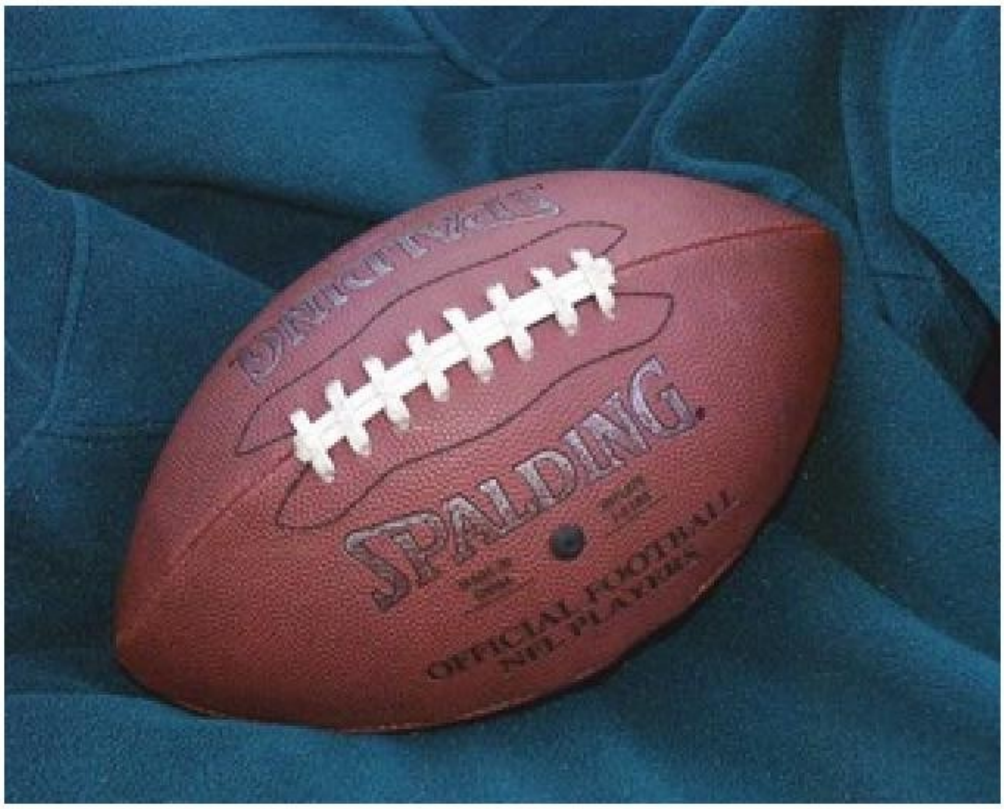
```
function im = blend(im1,im2,mask,level)
    im1 = im2double(im1);
    im2 = im2double(im2);

    mask = mask / 255;
    g = gaussian(mask,level);
    g = g{1};
    blend = cell(1,level);

    l1 = laplacian(im1,level);
    l2 = laplacian(im2,level);
    for i = level:-1:1
        temp = mat2gray(g{i});
        i1 = l1{i};
        i2 = l2{i};
        i2(:,:,1) = i2(:,:,1) .* temp;
        i2(:,:,2) = i2(:,:,2) .* temp;
        i2(:,:,3) = i2(:,:,3) .* temp;

        i1(:,:,1) = i1(:,:,1) .* (1 - temp);
        i1(:,:,2) = i1(:,:,2) .* (1 - temp);
        i1(:,:,3) = i1(:,:,3) .* (1 - temp);
```

```
        x = i1 + i2;
        blend{i} = x;
    end

    for i = level:-1:2
        t0 = blend(1,i);
        t1 = blend(1,i - 1);
        t0 = cell2mat(t0);
        t1 = cell2mat(t1);
        t0 = imresize(t0, [size(t1,1) size(t1,2)]);
        blend{i-1} = t1 + t0;
    end

    imshow(blend{1});
end
```



1C)
Nearest

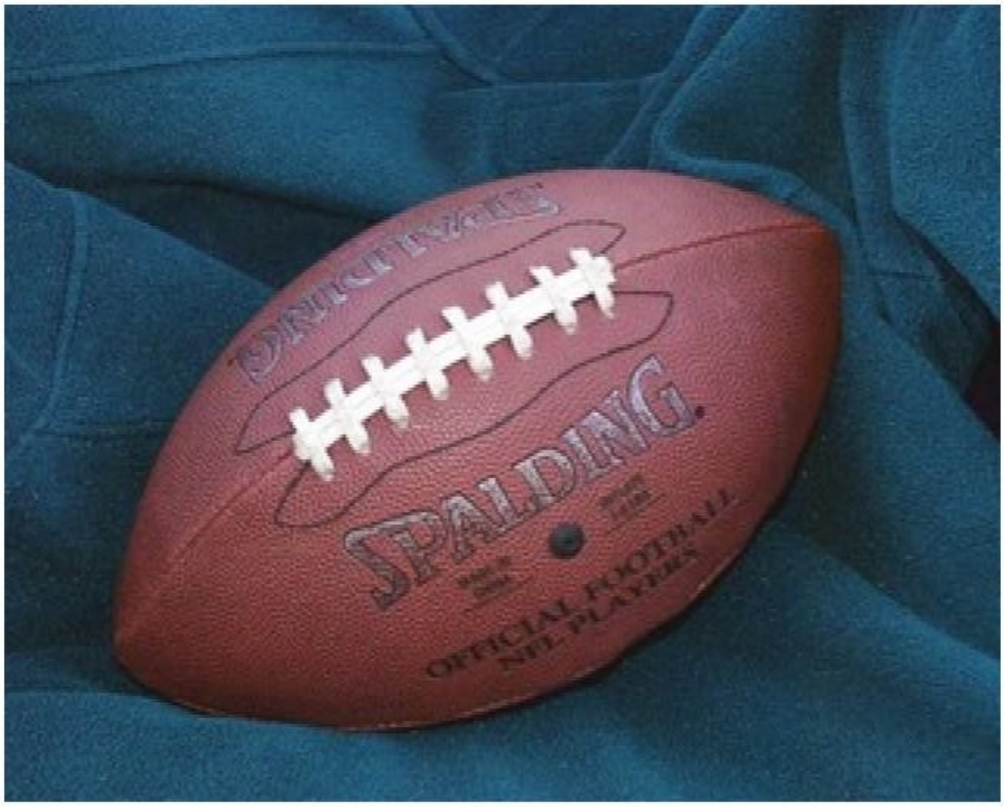
```
function oneNearest

img = imread('./Assign2_imgs/other_images/pears.png');

height = size(img, 1);
width = size(img, 2);
ht2 = height * 2;
wid2 = width * 2;

x = zeros(ht2, wid2,3);
x = x - 1;

x(1:2:ht2, 1:2:wid2,1) = img(1:height, 1:width,1);
x(1:2:ht2, 1:2:wid2,2) = img(1:height, 1:width,2);
x(1:2:ht2, 1:2:wid2,3) = img(1:height, 1:width,3);


for i = 1 : ht2
    for j = 1 : wid2
        if x(i, j, 1) == -1
            if mod(i, 2) == 0
                x(i, j, :) = x(i - 1, j,:);
            else
                x(i, j, :) = x(i, j - 1,:);
            end
        end
    end
end

img = x;

imshow(uint8(x));
end
```



Linear

```
img = imread('./Assign2_imgs/other_images/pears.png');

height = size(img, 1);
width = size(img, 2);
ht2 = height * 2;
wid2 = width * 2;

x = zeros(ht2, wid2,3);
x = x - 1;

x(1:2:ht2, 1:2:wid2,1) = img(1:height, 1:width,1);
x(1:2:ht2, 1:2:wid2,2) = img(1:height, 1:width,2);
x(1:2:ht2, 1:2:wid2,3) = img(1:height, 1:width,3);

for i = 1: 2: ht2
   for j = 1 : wid2
      if x(i, j,1) == -1
         if j == wid2
            x(i, j, :) = x(i, j - 1, :);
         else
            x(i, j,:) = floor((x(i, j - 1,:) + x(i, j + 1,:)) / 2);
         end
      end
   end
end

for i = 2: 2: ht2
   for j = 1: wid2
```

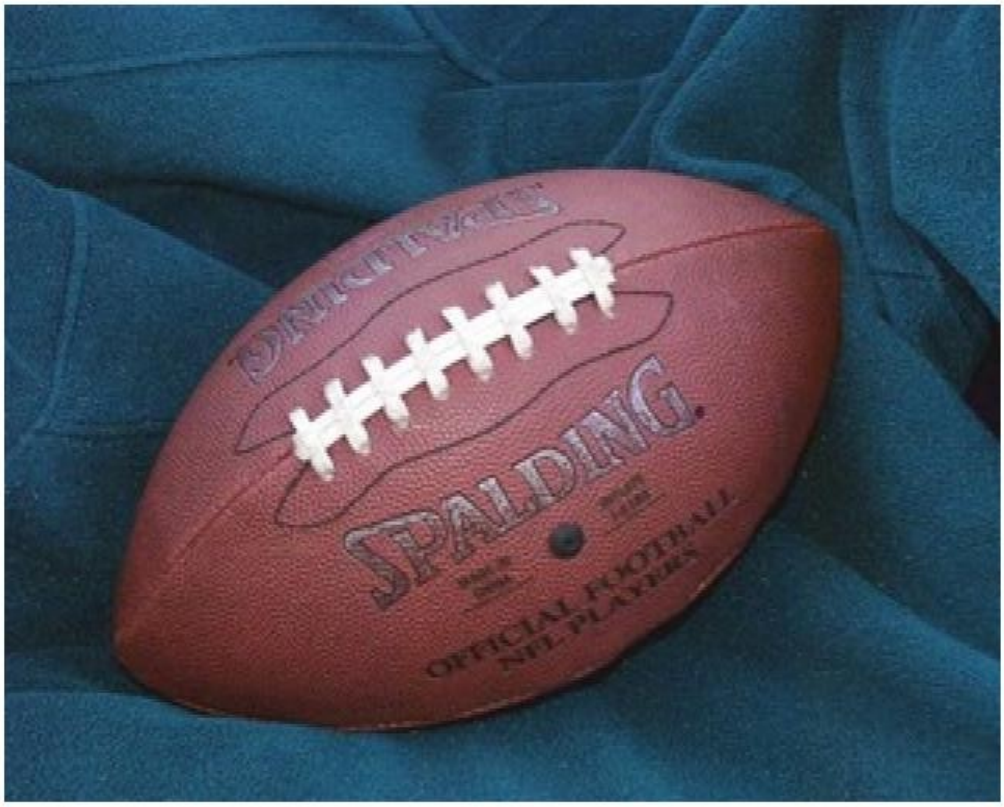
```
        x(i, j,:) = x(i - 1, j,:);
    end
end

imshow(uint8(x));
```



Bilinear

```
img = imread('./Assign2_imgs/other_images/pears.png');

height = size(img, 1);
width = size(img, 2);
ht2 = height * 2;
wid2 = width * 2;

x = zeros(ht2, wid2,3);
x = x - 1;

x(1:2:ht2, 1:2:wid2,1) = img(1:height, 1:width,1);
x(1:2:ht2, 1:2:wid2,2) = img(1:height, 1:width,2);
x(1:2:ht2, 1:2:wid2,3) = img(1:height, 1:width,3);

for i = 1: 2: ht2
    for j = 1: wid2
        if x(i, j, 1) == -1
            if j == wid2
                x(i, j, :) = x(i, j - 1, :);
            else
                x(i, j, :) = floor((x(i, j - 1, :) + x(i, j + 1, :)) / 2);
            end
        end
```

```
    end
end

for i = 2: 2: ht2
    for j = 1: wid2
        if i == ht2
            x(i, j, :) = x(i - 1, j, :);
        else
            x(i, j, :) = floor((x(i - 1, j, :) + x(i + 1, j, :)) / 2);
        end
    end
end

img = x;

imshow(uint8(x));
```



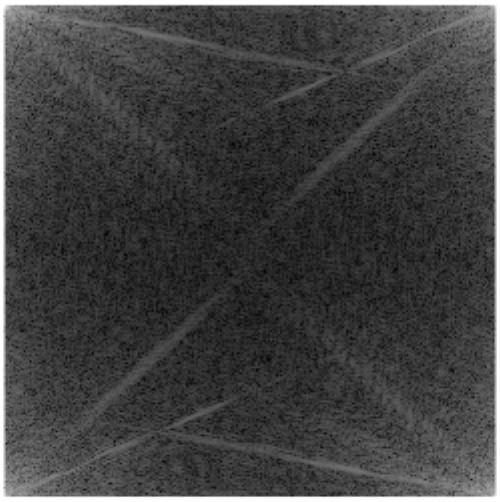
Bicubic

```
img = imread('./Assign2_imgs/other_images/football.jpg');

height = size(img, 1);
width = size(img, 2);
ht2 = height * 2;
wid2 = width * 2;

x = zeros(ht2, wid2,3);
x = x - 1;

x(1:2:ht2, 1:2:wid2,1) = img(1:height, 1:width,1);
```

```
x(1:2:ht2, 1:2:wid2,2) = img(1:height, 1:width,2);
x(1:2:ht2, 1:2:wid2,3) = img(1:height, 1:width,3);

for i = 1: 2: ht2
    for j = 1: wid2
        if x(i, j,1) == -1
            if j <= 4
                x(i, j,:) = floor((x(i, j - 1, :) + x(i, j + 1, :)) / 2);
            elseif j >= wid2 - 4
                x(i, j, :) = x(i, j - 1, :);

            else
                for c = 1:3
                    x(i, j, c) = floor(solver(x(i, j - 3, c), x(i, j - 1, c), x(i, j + 1, c), x(i, j + 3, c), j - 1, j +
1));
                end
            end
        end
    end
end

for i = 2: 2: ht2
    for j = 1: wid2

        if i <= 4
            x(i, j,:) = floor((x(i - 1, j,:) + x(i + 1, j,:)) / 2);
        elseif i >= ht2 - 4
            x(i, j,:) = x(i - 1, j,:);

        else
            for c = 1:3
                x(i, j,c) = floor(solver(x(i - 3, j,c), x(i - 1, j,c), x(i + 1, j,c), x(i + 3, j,c), i - 1, i + 1));
            end
        end

    end
end

dup = img;
dup(:) = 0;
x=uint8(x);
y = [dup,dup;dup,img];
imshow([y,x]);
```

2 A)

```
function [ ans ] = fft1d(arr, w)

n = size(arr,2);
half = n/2;

if n == 1
    ans = arr;
    return

else

  x = 1.0;
  ans = zeros(1,n);

  arr_odd = arr(1:2:n);
  arr_even = arr(2:2:n);

  ans_odd = fft1d(arr_odd, w*w);
  ans_even = fft1d(arr_even, w*w);


  for i = 1:half
      ans(i) = ans_even(i) + (x*ans_odd(i));
      ans(half + i) = ans_even(i) - (x * ans_odd(i));
      x = x * w;
  end

end

end

function [ ans ] = fft2d( path )

I = imread(path)
[R, C] = size(I);
im = im2double(I);
R = pow2(round(log2(R)));
C = pow2(round(log2(C)));
arr = imresize(im, [R C]);
temp = zeros(size(arr));
ans = zeros(size(arr));
for i = 1:R
    temp(i,:) = fft1d(arr(i,:), exp(-2*1i*pi/C));
end
for j = 1:C
    first = transpose(temp(:,j));
    second = transpose(exp(-2*1i*pi/R));
    ans(:,j) = fft1d(first, second);
end
final = mat2gray(log(1 + abs(ans)))
imshow(final);
end
```



FFT of
cameraman.tif

2 B)
Ideal

```
function [ imx ] = ideal(image, arg)
    radius = 10;
    im1 = rgb2gray(image);
    f = fftshift(fft2(im1));
    [H, W] = size(f);
    [x, y] = meshgrid(1:W, 1:H);
    c1 = floor(W/2);
    c2 = floor(H/2);
    dist = (x - c1).^2 + (y - c2).^2;
    filter = dist < radius^2;
    if arg == 1
        filter = dist >= radius^2;
    end

    f = f.*filter;
    imx = ifft2(ifftshift(f));
    imshow(uint8(abs(imx)));
end
```

Butterworth

```
function [ imx ] = butterworthHybrid(image,arg)
    im1 = rgb2gray(image);
    f = fftshift(fft2(im1));
    radius = 10;
    n = 5;
    [H, W] = size(f);
    [x, y] = meshgrid(1:W, 1:H);
    c1 = floor(W/2);
    c2 = floor(H/2);
    dist = (x - c1).^2 + (y - c2).^2;
    filter = 1 + (dist / radius^2) .^ n;
    if arg == 1
        filter = 1 + (radius^2 ./ dist) .^ n;
```

```
    end
    filter = 1 ./ filter;
    f = f.*filter;
    imx = ifft2(ifftshift(f));
    imshow(uint8(abs(imx)));
end
```

Gaussian

```
function [ imx ] = gaussianHybrid(image,arg)
    im1 = rgb2gray(image);
    f = fftshift(fft2(im1));
    sig = 10;
    [H, W] = size(f);
    [x, y] = meshgrid(1:W, 1:H);
    c1 = floor(W/2);
    c2 = floor(H/2);
    dist = (x - c1).^2 + (y - c2).^2;
    if arg == 0
        filter = exp(-1 * dist / (2 * sig^2));
    else
        filter = 1 - exp(-1 * dist / (2 * sig^2));
    end
    f = f.*filter;
```

```
    imx = ifft2(ifftshift(f));
    imshow(uint8(abs(imx)));
end
```

2 C)

```
I = imread('Assign2_imgs/other_images/concordaerial.png');
[m,n,d3] = size(I);
filt = ones(m,n);
for i=1:m
    for j=1:n
      x = (i-floor(m/2));
      y = (j-floor(n/2));
       filt(i,j) =  -(x^2 + y^2);
    end
end
fin = ones(size(I));
for i=1:3
    Ift = fftshift(fft2(I(:,:,i)));
    ans = Ift.*filt;
    Iift = ifftshift(ans);
```

```
    ans = abs(ifft2(Iift));
    myMax = max(-1*ans);
    myMax = max(myMax);
    ans = ans/myMax*255;
    fin(:,:,i)=I(:,:,i)+uint8(ans);

end
imshow(uint8(fin))
```

2 D)

```
I = imread('./Assign2_imgs/notch_pass_reject_filter/notch3.jpg');
I1 = im2double(I);
Ift = I1;
ansI = I1;
Ift(:,:,1) = fft2(I(:,:,1));
Ift(:,:,2) = fft2(I(:,:,2));
Ift(:,:,3) = fft2(I(:,:,3));

Ift(:,:,1) = fftshift(Ift(:,:,1));
Ift(:,:,2) = fftshift(Ift(:,:,2));
Ift(:,:,3) = fftshift(Ift(:,:,3));

Iift = mat2gray(log(1+abs(Ift)));

Ift(1:88,128:132,:) = 0;
Ift(170:256, 128:132,:) = 0;

Ift(128:132,1:88,:) = 0;
Ift(128:132,170:256,:) = 0;

ansI(:,:,1) = ifft2(ifftshift(Ift(:,:,1)));
ansI(:,:,2) = ifft2(ifftshift(Ift(:,:,2)));
ansI(:,:,3) = ifft2(ifftshift(Ift(:,:,3)));

Ift = mat2gray(log(1+abs(Ift)));
ansI=mat2gray(abs(ansI));
subplot(1,4,1); imshow(I);
```

```
subplot(1,4,2); imshow(Iift(:,:,1));
subplot(1,4,3);
imshow(Ift(:,:,1));
subplot(1,4,4);
imshow(ansI);
```