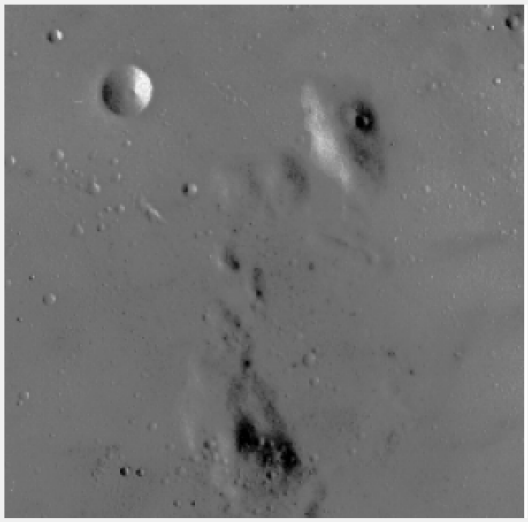
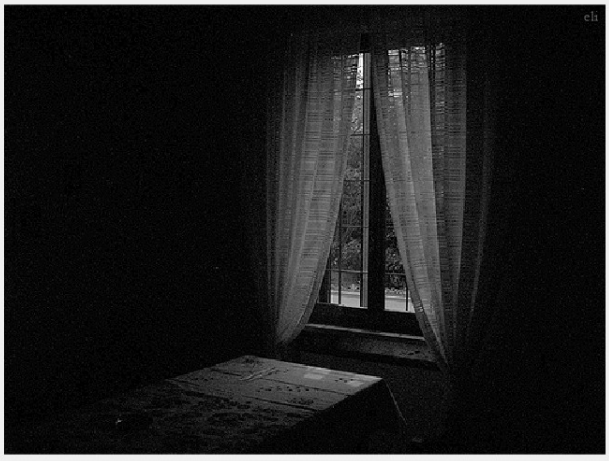
# HISTOGRAM MATCHING

## CODE:

```
clear all;
I = imread('hist_equal.jpg');
U = imread('hist_equal2.jpg');
counts1 = imhist(rgb2gray(I));
counts2 = imhist(rgb2gray(U));
mat = tril(ones(256,256),0);
cdf1=cumsum(counts1);
cdf2=cumsum(counts2);
x = cdf1(256,1);
y = cdf2(256,1);
for i = 1:256
    count3(i,1) = floor((cdf1(i,1)/x) * 256);
    count4(i,1) = floor((cdf2(i,1)/y) * 256);
end
i = 1;
trans = zeros(256,1);
for j = 1:256
    i=1;
    while (i<=256 && (count4(j,1)) >= (count3(i,1)) )
        trans(j,1) = i;
        i=i+1;
    end
end
I = rgb2gray(I);
image = trans(I(:,:)+1);
figure;
subplot(1,2,1);
imshow(I);
subplot(1,2,2);
imshow(uint8(image))
;
```



HISTOGRAM MATCHING



# 1. LOCAL HISTOGRAM EQUALISATION



## Code:

```
clear all;
im = imread('hist_equal.jpg');
im = rgb2gray(im);
no_of_bins = 256;
[x1,y1] = size(im);
mat_len = 151;
mat_wid = 151;
mid_val = round(mat_len*mat_wid/2);
mid_val_len = floor(mat_len/2);
mid_val_wid = floor(mat_wid/2);
padim = padarray(im, [mid_val_len,mid_val_wid]);
[x2,y2] = size(padim);
%imshow(padim);

for i=1:(x2-2*mid_val_len-1)
    for j = 1:(y2-2*mid_val_wid-1)

        inc=1;
        window = padim(i:i+mat_len-1,j:j+mat_wid-1);
        cdf =imhist(window);
        cdf=cumsum(cdf);
        newimg(i,j) = round(cdf(window(mid_val_len,mid_val_wid)+1)/(mat_len*mat_wid)*255);
% THE FOLLOWING CODE CAN BE UNCOMMENTED(Coded without Inbuilt Functions)

%         for k=0:mat_len-1
%             for l=0:mat_wid-1
%                 if(inc== mid_val)
%                     ele = padim(i+k-1,j+l-1)+1;
%                 end
%                     pixel = padim(i+k,j+l)+1;
%               %  l=l+1;
%                     cdf(pixel) = cdf(pixel) + 1;
%                     inc=inc+1;
%             end
%             %k=k+1;
%         end
%         %cumulative = zeros(1,256);
%         %display(cumulative);
%         %cumulative(1)=pdf(1);
%         for p=2:256
%             cdf(p) = cdf(p) + cdf(p-1);
%         end
%         %cumulative = cumulative./(mat_len*mat_wid);
```

```
%         %cumulative = cumulative.*256;
%         %cumulative = round(cumulative);
%         %newimg(i,j)=cumulative(padim(i+mid_val_len,j+mid_val_wid)+1);
%         newimg(i,j) = round(cdf(ele)/(mat_len*mat_wid)*255);
%         % j=j+1;
    end
    %i=i+1;
end
```

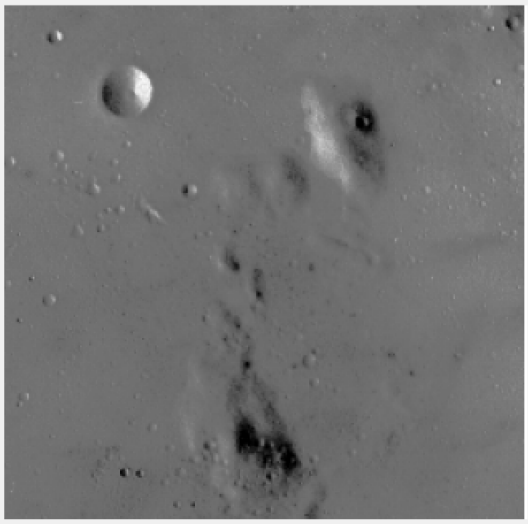
## Input Image

```
figure;
imshow(im);
```

## Output Image

```
imshow(uint8(newimg));
```

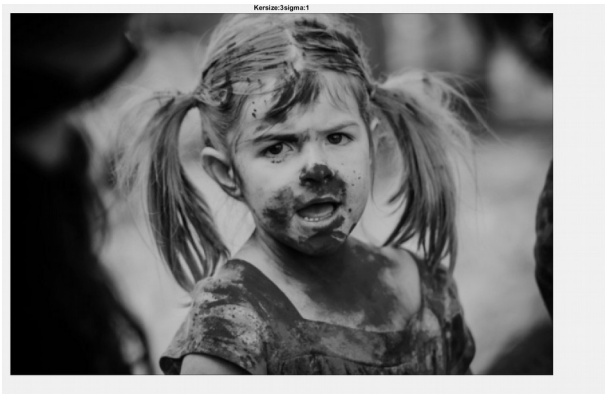
Published with MATLAB® R2015a

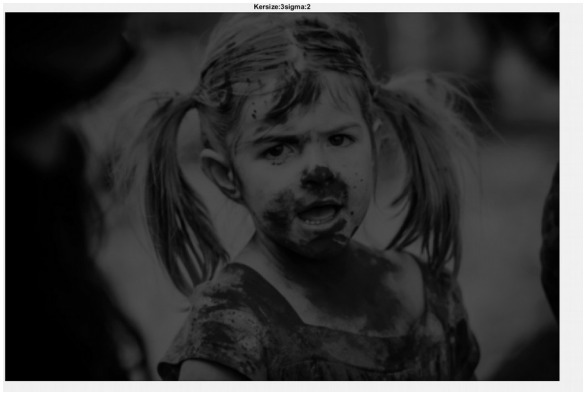


# GAUSIAN FILTER

## CODE:

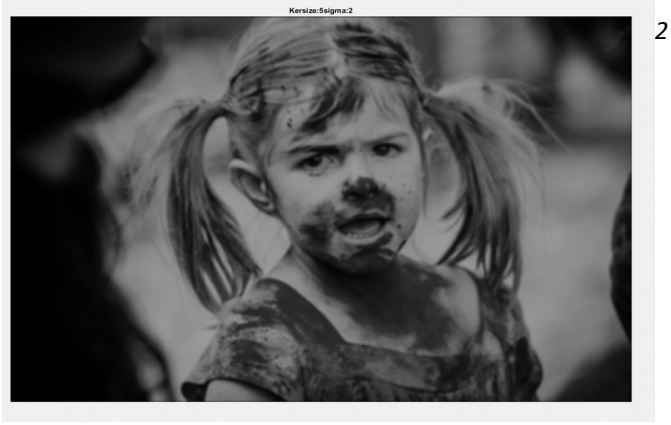
```
clear all;
for color=1:3
    im= imread('gauss.jpg');
    im=im(:,:,color);
    sigma = 2;
    %figure;
    %imshow(im);
    k = 1/(2*pi*sigma*sigma);
    %k=1;
    ker_size = 8;
    mid=floor(ker_size/2);
    window = zeros(ker_size,ker_size);
    [x,y]=size(im);
    for i=1:ker_size
        for j=1:ker_size
            window(i,j)= k*exp(-1*(((i-mid-1)^2+(j-mid-1)^2)(2*sigma*sigma)));
        end
    end
    %window = window.*265;
    %window = floor(window);
    padim =padarray(im,[mid,mid]);
    [x2,y2]=size(padim);
     for i = 1:x2-ker_size+1
        for j =1:y2-ker_size+1
            Temp = double(padim(i:i+ker_size-1,j:j+ker_size-1)).*window;
            final(i,j)=sum(Temp(:));
        end
     end
    %final = round(final./10000);
    %final = conv(im,window);
    figure;
    %subplot(1,2,1);
    %imshow(im);
    %subplot(1,2,2);
    %imshow(uint8(final));
    finalim(:,:,color)=uint8(final);
    %title(strcat('Kersize:', int2str(ker_size),'sigma: ',int2str(sigma)));
end
imshow(finalim);
```


Kernel size: 3  Sigma:1

Kernel size:3 sigma:2




2


Kernelsize:5 sigma:2

Kernelsize: 5 sigma:1


Kernel_size : 8
Sigma : 2


Kernel_size : 8
Sigma : 1

**NOTE :** CODE IS FOR COLORED IMAGES(can be modified for grayscale)

# MEDIAN FILTERING

## CODE:

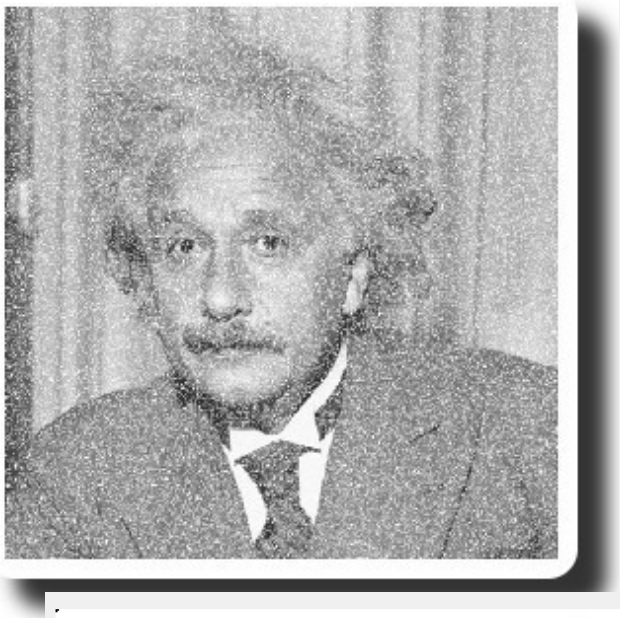
```
clear all;
for color=1:3
    im= imread('gauss.jpg');
    im=im(:,:,color);
    %figure;
   % imshow(im);
    ker_size =8;
    mid=floor(ker_size/2);
    padim = padarray(im,[mid,mid]);
    [x2,y2] = size(padim);
    for i=1:x2-2*mid-1
        for j=1:y2-2*mid-1
            window = padim(i:i+ker_size-1,j:j+ker_size-1);
            temp = sort(window(:));
            val = temp(round((ker_size*ker_size)/2));
            newim(i,j)=val;
        end
    end
    %figure;
    %imshow(newim);
    final(:,:,color)=newim;
end
figure;
imshow(final);
```

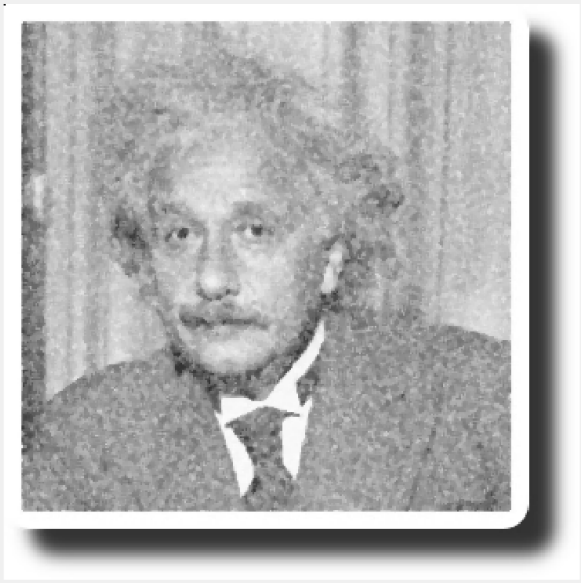
**INPUT IMAGE**

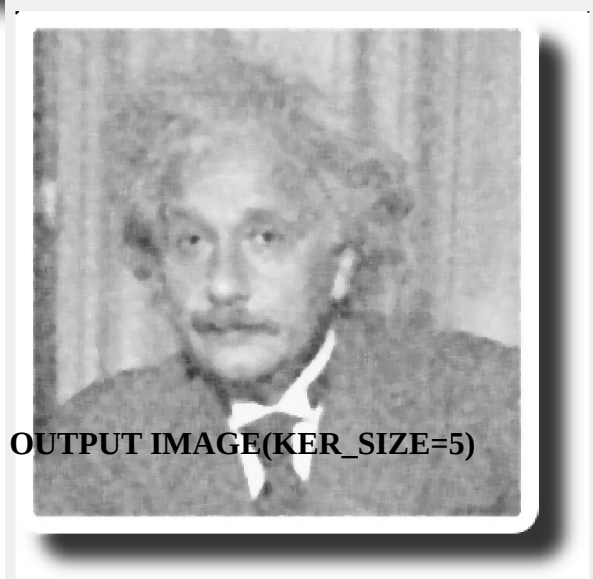
**OUTPUT IMAGE(KERNEL SIZE = 3)**

**OUTPUT IMAGE(KERNEL SIZE = 5)**

**OUTPUT IMAGE(KER_SIZE=8)**

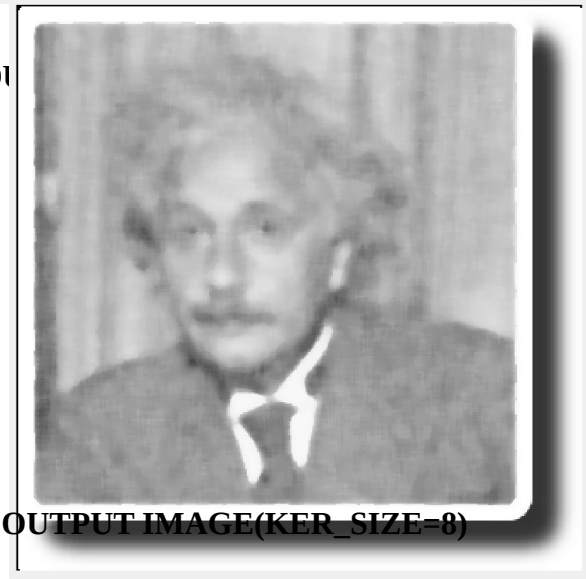






**OUTPUT IMAGE(KER_SIZE=5)**

**OU**



**OUTPUT IMAGE(KER_SIZE=8)**

# HIGH BOOST FILTERING

## CODE:

```
clear all;
for color=1:3;
    im= imread('blur2.jpg');
    im=im(:,:,color);
    sigma = 1;
    lambda = 2;
    figure;
    imshow(im);
    k = 1/(2*pi*sigma*sigma);
    %k=1;
    ker_size = 5;
    mid=floor(ker_size/2);
    window = zeros(ker_size,ker_size);
    [x,y]=size(im);
    for i=1:ker_size
        for j=1:ker_size
            window(i,j)= k*exp(-1*(((i-mid-1)^2+(j-mid-1)^2)/
(2*sigma*sigma)));
        end
    end
    %window = window.*265;
    %window = floor(window);
    padim =padarray(im,[mid,mid]);
    [x2,y2]=size(padim);
     for i = 1:x2-ker_size+1
        for j =1:y2-ker_size+1
            Temp = double(padim(i:i+ker_size-1,j:j+ker_size-1)).*window;
            gauss(i,j)=sum(Temp(:));
        end
     end
     subt = im - uint8(gauss);
    final =  im + lambda*subt;
    %final = round(final./10000);
    %final = conv(im,window);
    %figure;
    %imshow(uint8(final));
    newim(:,:,color)=uint8(final);
end
figure;
imshow(newim);
```

**NOTE :** Code is for colored images, can be modified for grayscale

Final_image = image + lambda *(image-gaussian)

**INPUT IMAGE**

**OUTPUT IMAGE**

**Lambda = 0.5**

**OUTPUT IMAGE(Lambda = 1)**

**OUTPUT IMAGE (Lambda = 2)**

# BILATERAL FILTERING

## CODE:

```
clear all;
pica = imread('zelda512-NoiseV400.jpg');
pic = im2double(pica); figure;
imshow(pica);
fsize = 8;
sigma = 2;
fsi = round(fsize/2)-1;
inds = -fsi : fsi;

picpad = padarray(pic, [fsi fsi]);
[ht,wid,pkh] = size(picpad);

ind = round(fsize / 2)-1;


for i = 1 + ind: ht - ind
    for j = 1 + ind: wid - ind
        box = picpad(i - ind: i + ind, j - ind: j + ind, :);

        [X Y] = meshgrid(inds, inds);
        gf = exp(-(X.^2 + Y.^2) / (2*sigma*sigma));
        for eachcolor=1:3
            dl(:,:,eachcolor)=box(:,:,eachcolor) -
 picpad(i,j,eachcolor);
        end

        dL = box(:,:,1) - picpad(i,j,1);
        H = exp(-(dl(:,:,1).^2+dl(:,:,2).^2+dl(:,:,3).^2)/
(2*sigma^2));

        for eachcolor=1:3
            temp(:, :, eachcolor) = box(:, :, eachcolor) .* H;
        end

        for eachcolor=1:3
            temp(:, :, eachcolor) = temp(:, :, eachcolor) .* gf;
        end

        for eachcolor=1:3
            temp(eachcolor) = sum(sum(temp(:, :, eachcolor)));
        end

        norm = gf .* H;
        norm = sum(sum(norm(:)));
        for eachcolor=1:3
            pic(i, j, eachcolor) = temp(eachcolor)/norm;
        end
```

```
    end
end
figure;
imshow(pic);
```

**INPUT IMAGE**

**OUTPUT IMAGE**

**F_SIZE=8**

**OUTPUT IMAGE**

**F_SIZE = 4**

**INPUT IMAGE**

**OUTPUT**

**F_SIZE=8**

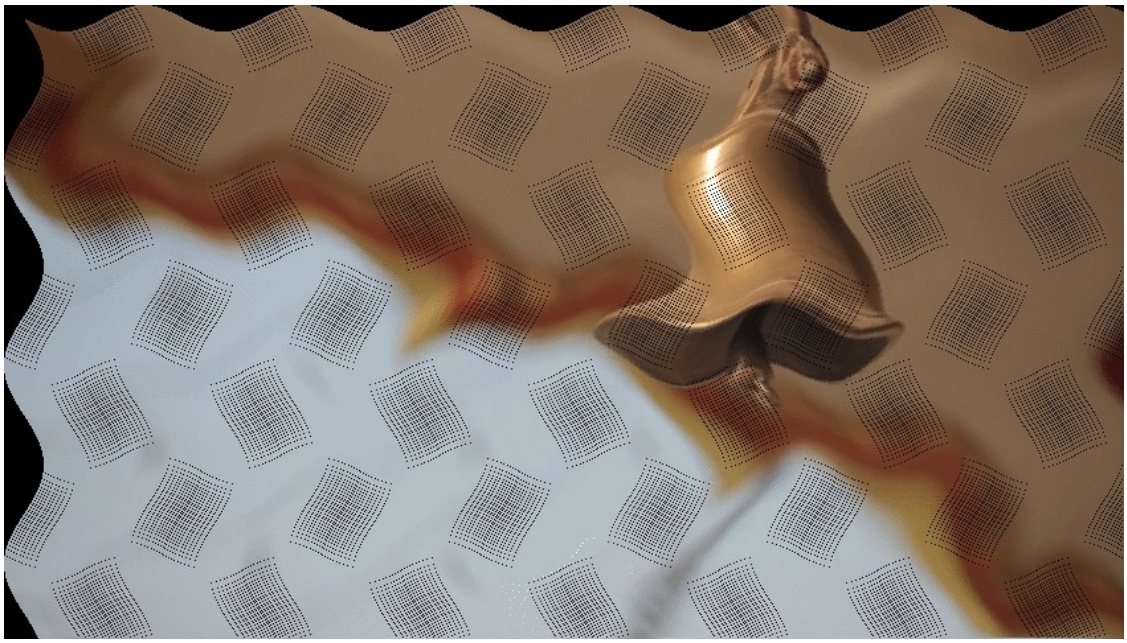
**INPUT IMAGE**

**OUTPUT IMAGE , F_SIZE = 8**

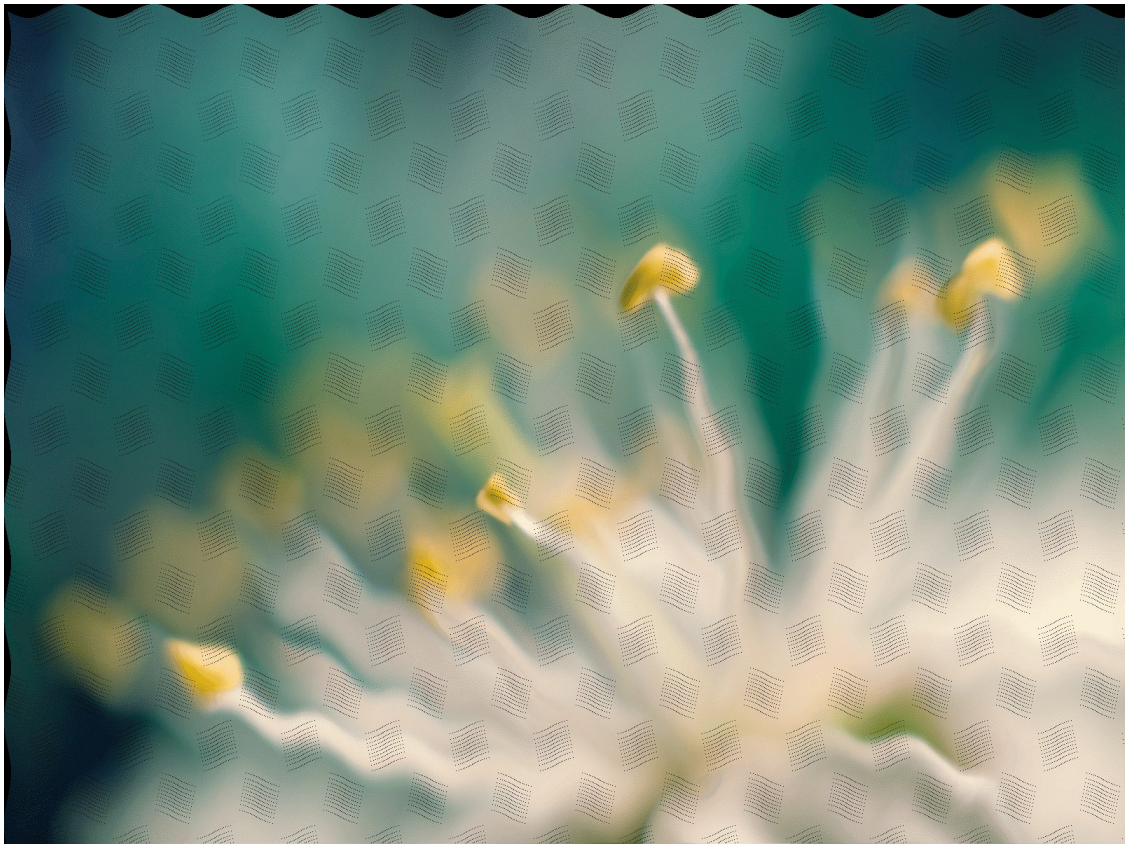
# RIPPLE TRANSFORM

## CODE:

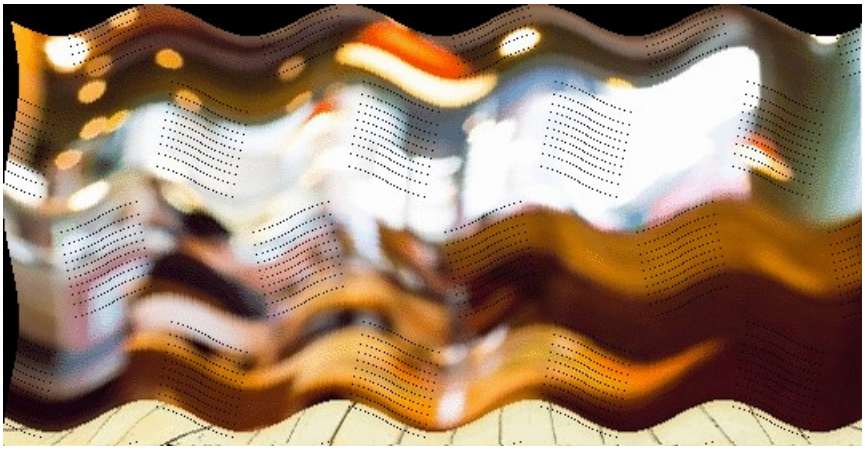
```
clear all;
im = imread('bell.jpg');
[x,y,cl]=size(im);
ax = 10;
ay = 15;
tx=120;
ty=150;
for p=1:25
    for i=1:x
        for j=1:y
            a  = i + ax* sin((2*pi*j)/tx);
            b = j + ay*sin((2*pi*i)/ty); %newim(round(a)
            +ax,round(b)+ay) = im(i,j);

            w = min(x,round(a)+ax);
            s = min(y,round(b)+ay);
            newim(w,s,1)=im(i,j,1);
            newim(w,s,2)=im(i,j,2);
            newim(w,s,3)=im(i,j,3);
        end
    end
    ax=ax+3;
    ay=ay+4;
    tx=tx+5;
    ty=ty+10;
    filename='ripple.gif';
    [A,map] = rgb2ind(uint8(newim),256);
    if p == 1
        imwrite(A,map,filename,'gif','LoopCount',Inf,'DelayTime',0.3);
    else

 imwrite(A,map,filename,'gif','WriteMode','append','DelayTime',0.3);
    end
end
 newim=uint8(newim);
imshow(newim);
```



**http://imgur.com/a/xkf5e**



**http://imgur.com/a/XThvU**



# SPHERICAL TRANSFORM

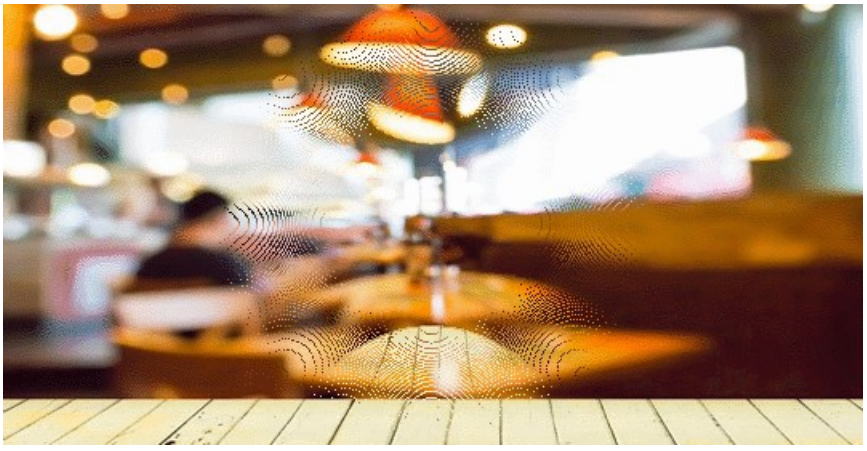
## CODE:

```
clear all;
im = imread('bell.jpg');
figure;
imshow(im);
[x,y,cl]=size(im);
xc = floor(x/2);
yc = floor(y/2);
rmax = floor(x/2);
r_index = 2.2;

for p=1:25
    newim = im;
    for i=1:x
        for j=1:y

            dx = i-xc;
            dy = j-yc;
            r= sqrt(dx*dx + dy*dy);
            z = sqrt(rmax^2 - r^2);
            betax=0;
            betay=0;
            if(r~=0 && z~=0)
                betax= (1-(1/r_index))*asin(dx/(sqrt(dx^2 + z^2)));
                betay = (1-(1/r_index))*asin(dy/(sqrt(dy^2 + z^2)));
            end
            a=i;
            b=j;
            if(r<=rmax)
                a = i-z*tan(betax);
                b = j-z*tan(betay);
            end
            w = min(x,round(a));
            s = min(y,round(b));
            newim(w,s,1)=im(i,j,1);
            newim(w,s,2)=im(i,j,2);
            newim(w,s,3)=im(i,j,3);
        end
    end
    r_index = r_index-0.07;
    rmax = rmax - 10;
    %imwrite('GIF','spherical.gif');
    %frame = getframe(1);
    %ima{p} = uint8(newim);
    filename='spherical.gif';
    [A,map] = rgb2ind(uint8(newim),256);
    if p == 1
        imwrite(A,map,filename,'gif','LoopCount',Inf,'DelayTime',0.3);
    else
```

```
 imwrite(A,map,filename,'gif','WriteMode','append','DelayTime',0.3);
    end
end
filename = 'testAnimatedspherical.gif'; % Specify the output file name
%for idx = 1:25

%end
newim=uint8(newim);
%imshow();
```

*Published with MATLAB® R2015a*

***http://imgur.com/a/KSIhK***



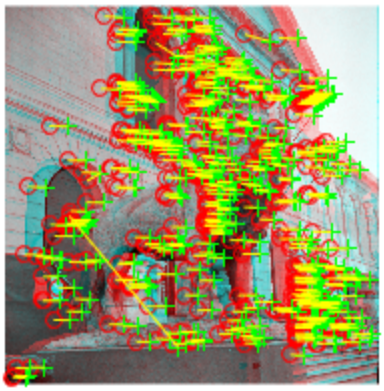
**http://imgur.com/a/Wo6wv**

**http://imgur.com/a/gYx11**

# STEREO MATCHING

## CODE:

```
im = imread('stereo_pair.jpg');
imsize = size(im);

firsthalf = im(:, 1:round(imsize(2) / 2)-1, :);
secondhalf = im(:, round(imsize(2) / 2):end, :);
firstgray = rgb2gray(firsthalf);
secondgray = rgb2gray(secondhalf);
firstfeatures = detectHarrisFeatures(firstgray);
[features1,valid_points1] = extractFeatures(firstgray,firstfeatures);
secondfeatures = detectHarrisFeatures(secondgray);
[features2,valid_points2] =
 extractFeatures(secondgray,secondfeatures);
indexPairs = matchFeatures(features1,features2);
matchedPoints1 = valid_points1(indexPairs(:,1),:);
matchedPoints2 = valid_points2(indexPairs(:,2),:);
figure;
subplot(1,2,1);
showMatchedFeatures(firstgray,secondgray,matchedPoints1,matchedPoints2);
match1 = matchedPoints1.Location;
[x,y]= size(match1);
match2 = matchedPoints2.Location;

[x,y] = size(match2);
match1 = reshape(match1, [2 x]);

match2 = reshape(match2, [2 x]);

v = homography_solve(match1, match2);

imp = secondhalf;
for i=1:size(firsthalf, 1)
    for j=1:size(firsthalf, 2)
        p = [i j 1]';
        ppn = v * p;
        ppn = floor(ppn ./ ppn(3))+1;
        for o=1:2
            ppn(o) = min(max(ppn(o), 1), size(secondhalf, o));
        end
        imp(ppn(1), ppn(2), :) = firsthalf(p(1), p(2), :);
    end
end
subplot(1,2,2);
imshow(imp - secondhalf);
```



*Published with MATLAB® R2015a*