HOMEWORK ASSIGNMENT:

To write an image filtering function and use it to create **hybrid images** using a simplified version of the SIGGRAPH 2006 **paper** by Oliva, Torralba, and Schyns.

INTRODUCTION:

A **hybrid image** is an image that is perceived in one of two different ways, depending on viewing distance, based on the way humans process visual input.
Hybrid images combine the low spatial frequencies of one picture with the high spatial frequencies of another picture, producing an image with an interpretation that changes with viewing distance.

CREATING HYBRID IMAGES :

Creating hybrid images is a two- step process.

STEP 1: Filter the 2 images using Gaussian filters to obtain the low frequencies from one image and the high frequencies from the second image.

STEP 2: The images as soon as they are filtered, they must be combined to obtain a hybrid image.

A) IMAGE FILTERING FUNCTION : *my_imfilter.m*

This file does the filtering operation using the following steps:

STEP 1: Detects the size of the given image and checks whether both the dimensions are odd or not.

STEP 1.1 : If the dimensions are not odd, then prints an error message and aborts the function

STEP 1.2 : If both the dimensions are odd, then proceed to step 2.

STEP 2: Get the size of the image along with the number of channels.

STEP 3: Assign the output with zeros of same image dimensions.

STEP 4: Check the number of channels of the image, to handle both color and grayscale images.

STEP 4.1 : If channel = 3, we filter each channel separately and combine them in the output.

STEP 4.2 : If channel = 1, we can directly pass the image to be filtered.

STEP 5: Helper function to filter the image - ***filterChannel(image,filter)***

STEP 5.1 : Get the size of the filter.

STEP 5.2 : Get the size of the image.

STEP 5.3 : Pre-allocate the size of the image for efficiency.

STEP 5.4 : Pad the input image with zeros. So that the filter goes through the image completely as the boundaries of image is reflected to create border. Uses Matlab 'padarray' function with 'symmetric' parameter is used to provide reflection for padding. Here, the resolution is preserved by padding with reflected image content.

STEP 5.5 : Iterate through each pixel of the image, pick the neighboring pixel and perform the convolution with the filter.

CODE SNIPPETS :

a) Filtering Operation :

b)

```
%Loop through rows and columns
%Iterate through each pixel of the image and
%pick the neighboring pixel
%To perform the convolution with the filter.
for i = 1:row
  for j = 1:col
    A = filter .* double(paddedImg(i:(i+(fr-1)), j:(j+(fc-1))));
    value = sum(sum(A));
    f_output(i,j) = value;
  end
end
```

b) Padding operation :

```
% Get the size of the filter
        [fr,fc] = size(filter);
% Pad the input image with zeros.
        paddedImg = padarray(image, [floor(fr/2), floor(fc/
2)],'symmetric');
```

c) Handling both color and grayscale images :

```
% The algorithm handles color images and grayscale images.
% if the image has three channels we filter each
% channel separately and combine them in the output.

if (channel == 3) % color image
    red = image(:,:,1);
    green = image(:,:,2);
    blue = image(:,:,3);

    % filtered channels are combined as the output.
    output(:,:,1) = filterChannel(red, filter);
    output(:,:,2) = filterChannel(green, filter);
    output(:,:,3) = filterChannel(blue, filter);

elseif (channel == 1) % grayscale image
    output = filterChannel(image, filter);
end
```

FILTERING ALGORITHM MEETS THE FOLLOWING REQUIREMENTS:

(1) Support grayscale and color images :

It supports both color as well as grayscale images. It checks for number of channel/ dimensions in the given image.

If the channel = 1 ; grayscale - then entire image is passed to the filterChannel function (helper function that performs the actual filter operation)

If the channel = 3; color image - then it handles colored images by filtering each of the color channels individually and then adding the channels back together.

(2) Support arbitrary shaped filters, as long as both dimensions are odd :

It displays an error message and aborts the function if both the dimensions are not odd. Proceeds otherwise.

(3) Pad the input image with zeros or reflected image content

It handles boundary condition by padding the input image with zeros using Matlab function ‘padarray’ using symmetric parameter to provide reflection.

(4) Return a filtered image which is the same resolution as the input image.

The resolution of the filtered image is preserved by padding with reflected image content.

B) HYBRID IMAGE GENERATION :

To combine two images into one hybrid image, we get the low frequencies from the first image and add to the higher frequencies of the second image.

Lower frequencies are obtained by filtering the image using Gaussian filter which removes frequencies beyond cut-off frequency. Thus, the output image includes only low frequencies.

Higher frequencies are obtained by subtracting the lower frequencies from the original image.

Then, combine both to get the hybrid image.

Code Snippet :

```
cutoff_frequency = 7;
filter = fspecial('Gaussian', cutoff_frequency*4+1,
cutoff_frequency);

low_frequencies =  my_imfilter(image1, filter);
high_frequencies = image2 - my_imfilter(image2, filter);
hybrid_image = low_frequencies+high_frequencies;
```

RESULTS :

A) FILTERING RESULTS :

Fig 1.1 : Original image : ***cat.bmp***

Fig 1.2 : Identity image : ***identity_image.jpg***

Fig 1.3 : Blur image : ***blur_image.jpg***

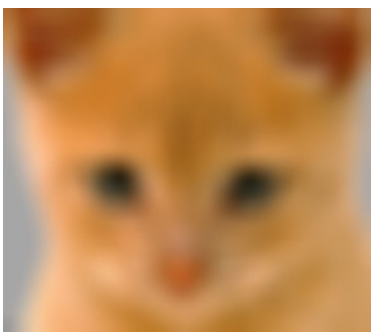


Fig 1.4 : Large Blur image : ***large_blur_image.jpg***

Fig 1.5 : Sobel image : ***sobel_image.jpg***

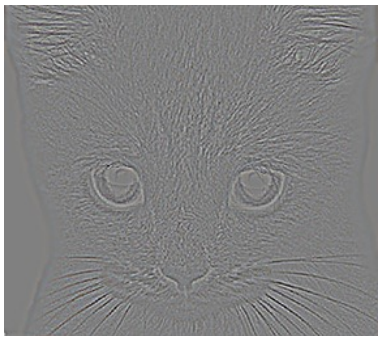


Fig 1.6 : Laplacian image : ***laplacian_image.jpg***

Fig 1.7 : High Pass image : ***high_pass_image.jpg***

B) HYBRID IMAGE RESULTS :

The cut-off frequency was tuned separately for each pair of images to produce the best results.

1) Dog & Cat : cut-off frequency : 7

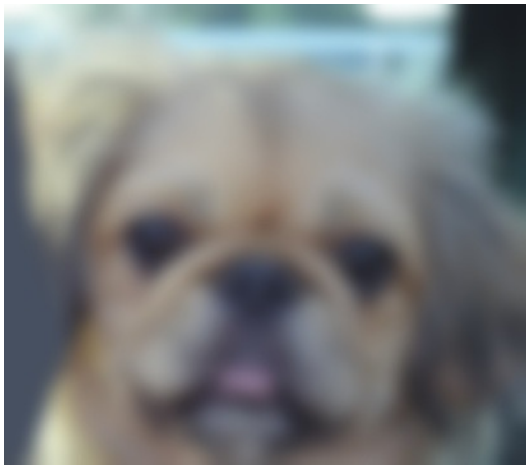


Fig 1.a low_frequencies.jpg

Fig 1.b high_frequencies.jpg

Fig 1.c hybrid_image.jpg

2) Bird & Plane : cut-off frequency : 3

Fig 2.a low_frequencies.jpg

Fig 2.b high_frequencies.jpg

Fig 2.c hybrid_image.jpg

3) Einstein & Marilyn : cut-off frequency : 4

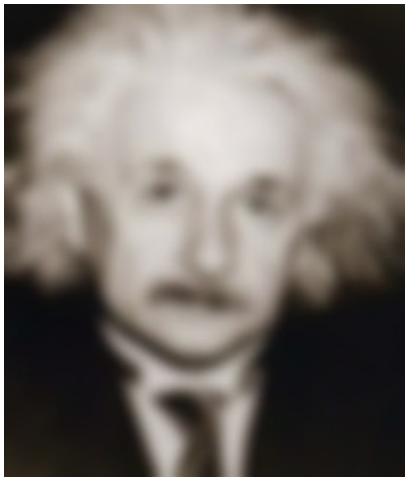


Fig 3.a low_frequencies.jpg

Fig 3.b high_frequencies.jpg

Fig 3.c hybrid_image.jpg

4) Fish & Submarine : cut-off frequency : 3

Fig 4.a low_frequencies.jpg

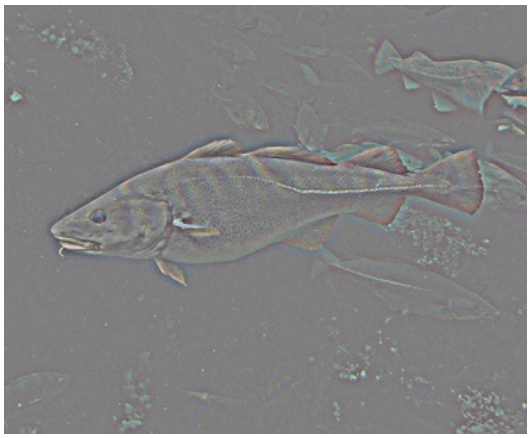


Fig 4.b high_frequencies.jpg

Fig 4.c hybrid_image.jpg

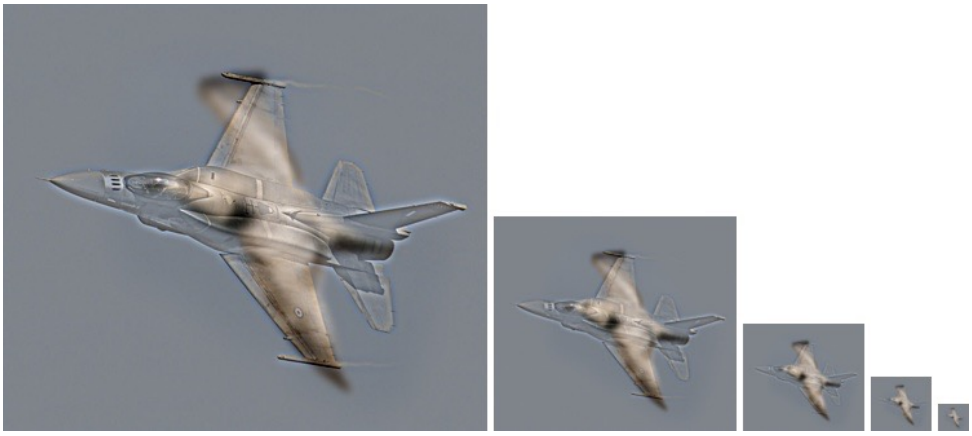
5) Bicycle & Motorcycle : cut-off frequency : 3

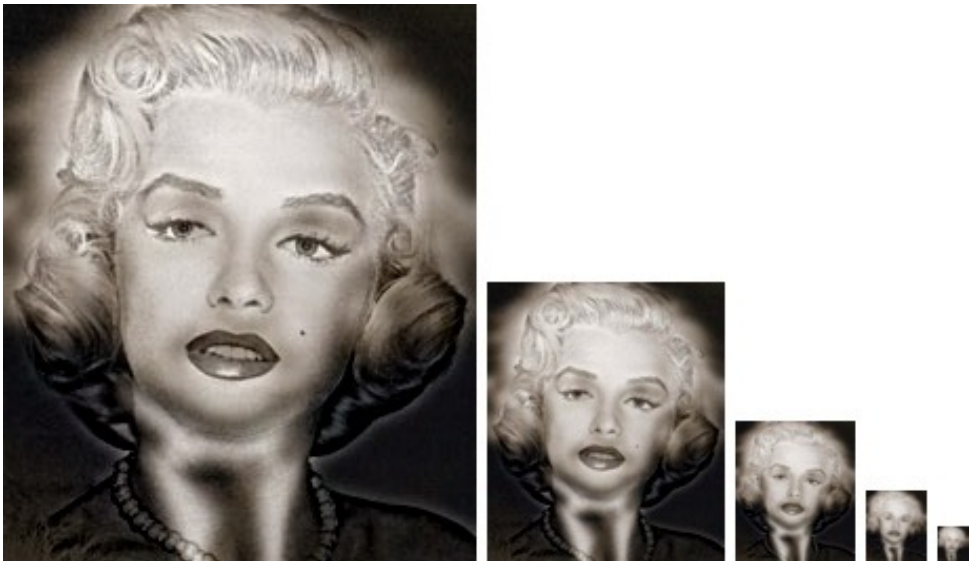
Fig 5.a low_frequencies.jpg

Fig 5.b high_frequencies.jpg

Fig 5.c hybrid_image.jpg

Scaled images :

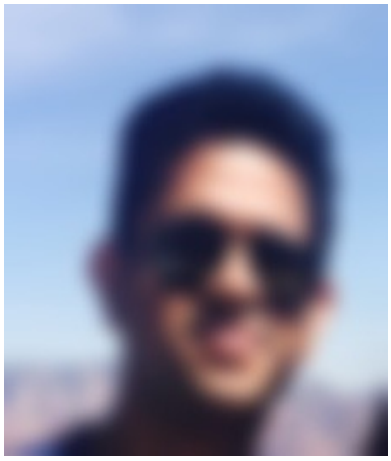




Extension : Hybrid image using Personal Images

Original images : ayish.bmp (left) shravan.bmp (right)


Filtered images : high_frequencies.bmp (left) low_frequencies.bmp (right)

Hybrid image :

Scaled Image :