Project 1: IMAGE FILTERING AND HYBRID IMAGES

**Project Prepared By:**

|  |  |
| --- | --- |
| Name | SAC STATE ID |
| MANSI PARESHKUMAR SONI | 302793530 |
| MEGHANA RAO KANNEGANTI | 302804645 |

**Introduction:**

This project can be divided into two main parts for simplification

* **Image Filtering:**

Image filtering is a type of convolution where a kernel is applied over each pixel and its local neighbors across the entire image. In this project we use the Gaussian filter to mainly blur the images to get its low frequencies and get the higher frequencies of another image by obtaining the difference of blurred image from the original image itself.

* **Hybrid Images:**

Hybrid Images are generated by merging two different images which vary with the viewing distances. We can observe that when the hybrid image is viewed closely, the high frequency image tends to dominate the view and going farther from the image, the low frequency image dominates the view.

**High Pass Filtered Image**

**Low Pass Filtered Image**

**Hybrid Images**

+

=

**Implementation:**

The my\_imfilter() function implemented here intends to replicate the behavior of MATLAB’s inbuilt function imfilter(). The objective of this function my\_imfilter() is to apply image filter on a given image layer by layer where the input image and filter are the inputs, and filtered image is the output with the same resolution and size as original input image. The my\_imfilter() function takes the input image and pads it with zeros in order to apply the filter on edges and applying the filter on the image by applying matrix multiplication layer by layer, iterating through the pixels of the padded image.

%Calclating the row and column margin, to pad the input image

pad\_row = (filter\_row - 1) / 2;  
pad\_col = (filter\_col - 1) / 2;  
%padding the input image  
for i = 1:size(image,3)  
padded\_img(:,:,i) = padarray(image(:, :, i), [pad\_row pad\_col]);  
end

%processing the output image

for k = 1:size(image,3)  
 for i = 1:size(padded\_img,1)-filter\_row+1  
 for j = 1:size(padded\_img,2)-filter\_col+1  
 %Calculate the output by convoluting the image and the filter  
 output(i,j,k) = sum(sum(filter.\*padded\_img(i:i+filter\_row-1,j:j+filter\_col- 1,k)));  
 end  
 end  
end  
end

**Obtaining the low and high frequency images:**

We are initializing the Gaussian filter with a custom cutoff\_frequency for different images and calling the my\_imfilter to apply the filter on the images to obtain the low frequencies from an image and then obtain the high frequency of another image by taking the difference between the original and low freq. image.

filter = fspecial('Gaussian', cutoff\_frequency\*4+1, cutoff\_frequency); low\_frequencies = my\_imfilter(image1, filter);

high\_frequencies = image2 - my\_imfilter(image2, filter);

**Executing code for Hybrid Image:**

Combining the images to obtain hybrid image

hybrid\_image = low\_frequencies + high\_frequencies;

**Results**

We have tried different images and get following output:

1. **Cat and Dog Image:**

A close up of a cat

Description automatically generated with medium confidence

A close up of a dog

Description automatically generated with medium confidence

A collage of a cat

Description automatically generated with medium confidence

1. **Bird and Jet Image:**

A bird flying in the sky

Description automatically generated with medium confidence

A model of a jet fighter

Description automatically generated with medium confidence

A close-up of a bug

Description automatically generated with low confidence

1. **Einstein and Marilyn Image:**

A picture containing text

Description automatically generated

A close up of a baby

Description automatically generated with low confidence

A picture containing text

Description automatically generated