Homework Assignment 2

CS696, applied computer vision

Instructor: Xiaobai Liu

Due: 11:59pm, Feb 24, 2017

Overview

The goal of this assignment is to write an image filtering function and use it to create [**hybrid images**](http://cvcl.mit.edu/hybridimage.htm) using a simplified version of the SIGGRAPH 2006 [**paper**](http://cvcl.mit.edu/publications/OlivaTorralb_Hybrid_Siggraph06.pdf) by Oliva, Torralba, and Schyns.

**Part-I: hybrid image generation in spatial domain**

*Hybrid images* are static images that change in interpretation as a function of the viewing distance. The basic idea is that high frequency tends to dominate perception when it is available, but, at a distance, only the low frequency (smooth) part of the signal can be seen.

By blending the high frequency portion of one image with the low-frequency portion of another, you get a hybrid image that leads to different interpretations at different distances.



Fig. 1, hybrid image. Look at image on right from very close, then from far away

Details

This project is intended to familiarize you with MATLAB and image filtering. Once you have created an image filtering function, it is relatively straightforward to construct hybrid images.

*Image Filtering*

Image filtering (or convolution) is a fundamental image processing tool. See chapter 3.2 of Szeliski and the lecture materials to learn about image filtering (specifically linear filtering).

MATLAB has numerous built in and efficient functions to perform image filtering, but you will be writing your own such function from scratch for this assignment. More specifically, you will implement my\_imfilter() which imitates the default behavior of the build in imfilter() function.

As specified in my\_imfilter.m, your filtering algorithm must

(1) support grayscale and color images

(2) support arbitrary shaped filters, as long as both dimensions are odd (e.g. 7x9 filters but not 4x5 filters)

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

(4) Return a filtered image which is the same resolution as the input image. We have provided a script, proj2\_test\_filtering.m, to help you debug your image filtering algorithm.

*Hybrid Image*

A hybrid image is the sum of a low-pass filtered version of the one image and a high-pass filtered version of a second image. There is a free parameter, which can be tuned for each image pair, which controls *how much* high frequency to remove from the first image and how much low frequency to leave in the second image. This is called the "cutoff frequency".

In the paper it is suggested to use two cutoff frequencies (one tuned for each image) and you are free to try that, as well.

In the **starter code**, i.e., proj2.m, the cutoff frequency is controlled by changing the standard deviation of the Gaussian filter used in constructing the hybrid images.

We provide you with 5 pairs of aligned images, in the direction of data, which can be merged reasonably well into hybrid images. The alignment is important because it affects the perceptual grouping (read the paper for details).

We **encourage** you to create additional examples (e.g. change of expression, morph between different objects, change over time, etc.).

For the example shown in Fig.1 , the two original images look like the images in Fig.2.



Fig. 2 The low-pass (blurred) and high-pass versions of these images look like this:



Fig 3. The high frequency image is actually zero-mean with negative values so it is visualized by adding 0.5. In the resulting visualization, bright values are positive and dark values are negative.

Adding the high and low frequencies, shown in Fig.3, together gives you the image in Fig.1 . If you're having trouble seeing the multiple interpretations of the image, a useful way to visualize the effect is by progressively down-sampling the hybrid image as is done below:



Fig. 4 The starter code provides a function vis\_hybrid\_image.m to save and display such visualizations.

*Potentially useful MATLAB functions:*

fspecial() and the operators padarray() in the [**MATLAB tutorial**](http://cs.brown.edu/courses/cs143/docs/matlab-tutorial/) which make it efficient to cut out image subwindows and do the convolution (dot product) between them.

*Forbidden functions*

you can use for testing, but not in your final code: imfilter(), filter2(), conv2(), nlfilter(), colfilt().

Write-up

In the report you will describe your algorithm and any decisions you made to write your algorithm a particular way. Then you will show and discuss the results of your algorithm. In the case of this project, show the results of your filtering algorithm (the test script saves such images already) and show some of the intermediate images in the hybrid image pipeline (e.g. the low and high frequency images, which the starter code already saves for you).

**Part-II: hybrid image generation in frequency domain**

In the above part I , you have been asked to implement a low-pass filter in spatial domain, i.e. the function my\_filter().

In this problem set, you will implement the same function in the frequency domain.



*Figure 1. image filtering in frequency domain. top-left: input image; top-right: smoothed images (low-pass); bottom-left: original frequency spectrum; bottom-right: low-passed frequency spectrum. Source codes: demo\_lpfilter.m*

**Review of image filtering in frequency domain**

Given an image I, the low-pass image filtering in frequency domain includes the following steps:

Step 1, apply the Fourier transform over I, i.e., fft2(I);

Step 2, remove the high-frequency components, e.g., by multiplying fft2(I) with a low-pass filter, e.g. Gaussian;

Step 3, apply the inverse Fourier transform to get the filtered image.

Figure 1 plots the original image, filtered images, original frequency *spectrum* and the filtered frequency *spectrum* (clock-wise).

The same pipeline applies to the high-pass filtering.

**Coding Requirements**

* Rewrite the function my\_filter() to implement both the low-pass filtering and high-pass filtering in frequency domain; add a flag variable in the parameters to indicate which function to call.
* Support gray-scale images only;
* In your project HA2, call my\_filter() to get the hybrid image for ONE pair of images (no extra credits for more pairs).
* Use TWO different low-pass thresholds and compare the results.

**Useful functions**

* Lpfilter(), attached with this homework assignment as well. Please check the demo\_lpfilter.m and the lpfilter.m;
* fft2()
* ifft2()
* fftshift();

**Forbidden functions**

None

**Write-up**

In the report you will describe your algorithm and any decisions you made to write your algorithm a particular way. Then you will show and discuss the results of your algorithm.

In the case of this problem set, show the hybrid images you got. Show the visualization of intermediate steps (like figure 1).

Also, discuss anything extra you did. Feel free to add any other information you feel is relevant.

How to submit

* Submit your source codes and writeup through SDSU Blackboard, before 11:59pm, Feb 24, 2017
* Three submissions are allowed.