EE5178 - Modern Computer Vision
Programming Assignment 3

Filtering and Hybrid Images

Note:

1. For any questions, please schedule a time with TAs before deadline according to their convenience. Please use moodle dicussion threads for posting your doubts and also check it before mailing to TAs, if the same question has been asked earlier.
2. Submit a single zip file in the moodle named as PA2_Rollno.zip containing report and folders containing corresponding codes.
3. Read the problem fully to understand the whole procedure.
4. Comment your code generously.
5. Put titles to all of the plots that you will show. Also label the x-axis and y-axis.
6. Save your iPython notebook as an html file with all the plots and results and submit it along with the iPython notebook containing your code.
7. In case of any doubts, students are encouraged to use the forum in the course moodle page.

---

1 Filtering

In signal processing, filtering is a process of removing unwanted features or components from a signal. Most often, this means removing or suppressing some frequencies or frequency bands. Filtering operation can be linear, non-linear, space-variant or invariant.

Below, we will look at some operations that can be done on a signal. For the below operations, consider a discrete time, 1D signal X of length 16, where $X = \{x_0, x_1, x_2, \cdots, x_{15}\}$. The output of the filtering process is a signal $Y = \{y_0, y_1, y_2, \cdots, y_{15}\}$, the same length as X. Implement the following filtering operations on the signal X defined as $x_k = 3 + sin 2\pi k/15$ for $k = \{0, 1, ..., 15\}$ and 0 otherwise.

For the filters that are linear and space-invariant, verify that the convolutional implementation of the filter gives the same output as direct implementation.

a) $y_k = x_{k+1} - x_k$

b) $y_k = x_k - \bar{X}$, where $\bar{X} = \frac{1}{L+1} \sum_{i=0}^{L} x_i$

c) $y_k = median\left(\{x_l : l \in [k-2 : k+2]\}\right)$

d) $y_k = x_{k+0.5} - x_{k-0.5}$ (Note: Linearly interpolate the neighboring samples to obtain signal values such as $x_{k+0.5}$.)

e) $y_k = |x_{k+0.5} - x_{k-0.5}|$

f) $y_k = \frac{1}{5} \sum_{i=k-2}^{k+2} x_i$

Tasks

1. Implement each of the filtering operations to obtain the desired output. Each of the outputs have to be the same size as the input signal.

2. For each operation, determine if these operations are linear and space-invariant.

3. Those operations that are linear and space-invariant, propose an equivalent convolution operation to implement the filtering process and also implement it.

4. For filters that are implemented via convolution, verify if the results are the same visually.

2 Filtering in Fourier Space

Filtering operations that are linear and space-invariant can be represented as a convolutional operation. Once, such a representation is obtained, we can use the Fourier property of convolutions to efficiently implement the filter in the Fourier domain.

Tasks

1. For those filters above that are linear and space-invariant, implement them in the Fourier domain.
2. Verify that the desired output from the Fourier implementation is the same as the spatial domain implementation. If there's any difference, explain why.
3. If for any of the cases, the output from the spatial and Fourier domain implementations are different, then suggest the modification to make the outputs same. Implement the modification and re-verify if the results are the same.

3 Hybrid Images

We will write an image convolution function (image filtering) and use it to create hybrid images. The technique was invented by Oliva, Torralba, and Schyns in 2006, and published in a paper at SIGGRAPH. High frequency image content tends to dominate perception but, at a distance, only low frequency (smooth) content is perceived. By blending high and low frequency content, we can create a hybrid image that is perceived differently at different distances. A hybrid image is the sum of a low-pass filtered version of a first image and a high-pass filtered version of a second image. We must tune a free parameter for each image pair to control 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 cut-off frequency. The paper suggests to use two cut-off frequencies, one tuned for each image, and you are free to try this too. Using a single cut-off frequency for both images should be sufficient. We will use a symmetric, zero-mean Gaussian filter for our filtering (low-pass and high-pass) operations. In our case, the cut-off frequency will represent the standard deviation of the Gaussian filter that will be used.

Process to generate a hybrid image:

(a) Implement a *my_filter.py* function to implement the 2-D image filtering operation. Your function should:

- pad the input image with zeros before filtering.
- accept any arbitrary filter kernel with which to convolve the image. If the filter has even dimension, then raise an exception.
- Return a filtered image of the same spatial resolution as the input.
- Supports filtering of both grayscale and colour images.

(b) Implement a function to generate a Gaussian kernel of a given standard deviation. The standard deviation represents the cut-off frequency of the filter.

(c) Remove high frequencies from image1 by blurring image1 with the Gaussian filter.

(d) Remove low frequencies from image2 using a two-step process. First, remove high frequencies from image2 using the process described in (c). Then subtract the low-pass filtered image2 from the original image to leave only the high frequency components.

(e) Each of the filtered images can have image values that are smaller than 0.0 or larger than 1.0. In such cases the clip the values smaller than 0.0 to 0.0 and values larger than 1.0 to 1.0. For high pass filtered image add a constant value of 0.5 to the whole image before clipping the values to be between 0.0 and 1.0.

(f) Combine the two images to generate the hybrid image.

Once the hybrid images are successfully created, you can view it from different distances to perceive different images. A useful way to visualize this hybrid image is by progressively downsampling the image. By progressive downsampling, we remove a part of the frequency content of the signal. Which part of the frequency content is removed and why? How does this affect the visualization of the hybrid images at different resolutions?

Tasks

1. There are 7 different pairs of images in the data directory provided. For each pair, you have to generate 2 different hybrid images by considering the first image in the directory as image1 and image2 respectively. These pairs of images can be color, or grayscale and also of different resolutions. Your code should be able to handle all the different cases.

2. Use the function provided in the *helpers.py* file to visualize all the hybrid images at different resolutions.

3. Don't forget to tune the cut-off frequency for each pair of hybrid images to get the most visually pleasing results.