CS303B: Lab 3

Aim: To investigate the Fourier transform and different edge detection approaches using Matlab.

Fourier transform and filtering

1. Load the lena.bmp image
2. Use *fft2*() to compute its Fourier transform.
3. Use *fftshift*() to shift the Fourier transform and make it symmetric, denoted at IF
4. Compute the power spectrum and phase, and visualize them
5. Design an ideal low pass filter with R = 5
6. Filter IF, and then use FFT to shift it back
7. Use *ifft2*() to compute the inverse Fourier transform,
8. Visualize the filtered image

Gaussian Smoothing

Now implement a Gaussian filter method. You'll probably want to use a 7x7 mask. The sigma could normally be set as $\sigma = w/6$, where w is the window size. This constrain is to ensure that the window size is large enough to cover more that 98% of the Gaussian distribution profile. Convolving the image 'lena.bmp' with the mask should smooth it (low pass filter). Though Matlab had a build-in function *fspecial*() to create filter mask, to have a better understanding, you should write the code yourself to create the filter mask, and then call *conv2()* to perform filtering with zero padding .

Test your implementation.

Edge Enhancement

1. Implement a method that performs gradient-based edge enhancement based on the **Sobel** operator. It should convolve (you should use the *conv2* function)the input image with Sobel filter masks and then compute the gradient magnitude image. Test your method.
2. Implement a method that performs gradient-based edge enhancement based on the following gradient operator $[1, -1]$ and its transpose$[1, -1]'$. It should convolve the input image with these filter masks and then compute the gradient magnitude image. Test your method.

Simple Edge Detection

Use the Gaussian and edge enhancement methods you have written to perform edge detection. Experiment with different amounts of smoothing (i.e. apply the Gaussian filter more than once) and different thresholds. Can you describe how varying these parameters affect the result?

Canny Edge Detection

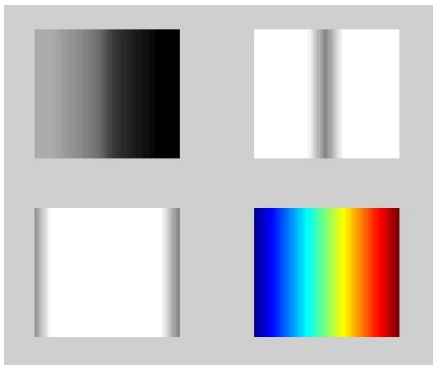
Using the function *edge*( ) (e.g., BW = EDGE(I,'canny',THRESH,SIGMA)) to detect edges in the image 'lena.bmp'. Can you describe how varying the thresholds [low, high] and Gaussian width affect the result?

Try Color Space (Matlab Example):

Load image *peppers.png* using *imread('peppers.png');* (matlab Build-in Image), use the function *rgb2hsv()*to converts RGB images to the HSV color space. *hsv2rgb* performs the reverse operation. These commands convert an RGB image to the HSV color space.

1) Display the R, G, B channels individually with each is treated as greyscale images
2) Display the H, S, V channels individually with each is treated as greyscale images

Using the following code to create an RGB image stored in the RGB variable
*RGB=reshape(ones(64,1)*reshape(jet(64),1,192),[64,64,3]);*
Convert it into an HSV image *HSV=rgb2hsv(RGB);* Display the H, S, V channels individually. Compare each channel to the RGB images; this could give you a closer inspection of the HSV color space, and its link to the RGB space. The H,S,V channels could be displayed as follow:



More comments on this example from the last paragraph from:

https://www.mathworks.com/help/images/understanding-color-spaces-and-color-space-conversion.html

Try Optical Flow (Matlab Example):

Try the example of Optical Flow from:

https://www.mathworks.com/help/imaq/examples/live-motion-detection-using-optical-flow.html

(Before trying the example, please install the package "Image Acquisition Toolbox" and "Image Acquisition Toolbox Support Package for OS Generic Video Interface". https://ww2.mathworks.cn/matlabcentral/fileexchange/45183-image-acquisition-toolbox-support-package-for-os-generic-video-interface, https://ww2.mathworks.cn/help/imaq/installing-the-support-packages-for-image-acquisition-toolbox-adaptors.html)

And :

1) Display the horizontal optical flow and vertical optical flow respectively
2) Display the flow magnitude image
3) Try to do some thresholding on the flow magnitude image, to see if you can get some clues to segment the moving object.

Finally…

Backup your project so that you might continue its development next week.