Digital Image Processing: Lab Assignment 3

#1: Image Filtering and Scaling [100points]

Issue date: -01-2020 Due date: -01-2020

Instructions

- Do not copy code from any other source (internet or friend). In case, any plagiarism detected, strictly zero mark will be assigned for that assignment.
- Show your results on sample images given in the assignment. Any other won't be considered for evaluation.
- Clearly state your name and Entry number on the lab report.
- Any additional materials used during the completion of the assignment must be cited. Failure to correctly reference sources will result in mark deduction(-10p/day).
- Submit a PDF file with proper. If the report is handed in more than three days after the deadline, the assignment will be marked zero marks. Up to fifty bonus points may be awarded to the student for very good lab assignments that comply with the criteria described below:

  +10p ← Report is clearly written and easy to follow.
  +10p ← Code is well documented.
  +10p ← Explanations and Observations are well written.
  +10p ← For overall exceptional reports, that confirm to all scientific writing standards.
  +10p ← Extra experiments performed on other set of images for better understanding.

Image Filtering[50]

1) In this problem, you will analyze the effect of an IIR filter specified by a 2-D difference equation. Let h(m, n) be the impulse response of an IIR filter with corresponding difference equation
$y(m, n) = 0.01x(m, n) + 0.9(y(m - 1, n) + y(m, n - 1)) - 0.81y(m - 1, n - 1)$
where x(m, n) is the input and y(m, n) is the output image.

   (i) Use Matlab to compute the point spread function of the filter, h(m, n), by applying the above difference equation to a 256×256 image of the form x(m, n) = δ(m−127, n−127). Export the result to a TIFF file using the following scaling, imwrite(uint8(255*100*h),’h_out.tif’) [15]

   (ii) Modify the program Example so that it filters the red, green and blue components of img.tif with the filter h(m, n) and generates a full color output image. [10]

2) In this problem, you will analyze the effect of a sharpening filter known as an unsharp mask. The terminology comes from the fact that an unsharp mask filter removes the unsharp (low frequency) component of the image, and therefore produces an image with a sharper appearance.
Let I(m,n) is the input image, given below and h(m, n) be a low pass filter. For our purposes use

$$h(m,n) = \begin{cases} 1/25\, for\, |m| \le 2 and\, |n| \le 2 \\ 0 otherwise \end{cases}$$

The unsharp mask filter is then given by
$g(m,n) = \delta(m,n) + \lambda(\delta(m,n) - h(m,n))$
where λ is a constant greater than zero.

   (i) Plot the output image y(m,n) after applying h(m,n) to input image I(m,n). Apply filter to each channel red, green and blue, at last generates the full color output image(RGB) “imgblur.tif”. [10]

   (ii) Apply your sharpening filter to “imgblur.tif” for λ = 0.2, 0.8, 1.5. Use Matlab, or some other image display tool, to view and compare each result to the original image. [15]

Fourier Transform[30]

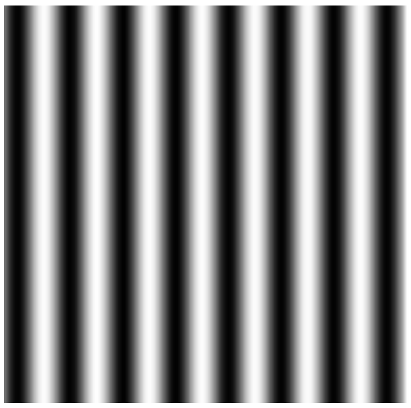
3. Create a 500x500 image of vertical black and white bars of equal width.

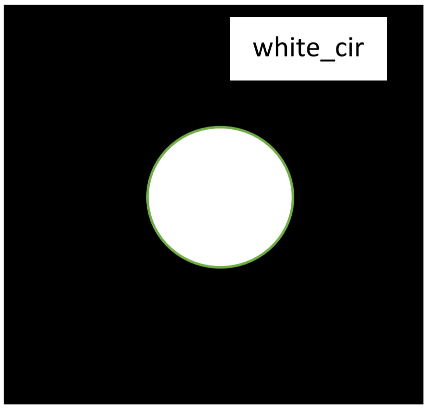
(i) Find its Fourier transform (plot both real and imaginary components). [5]

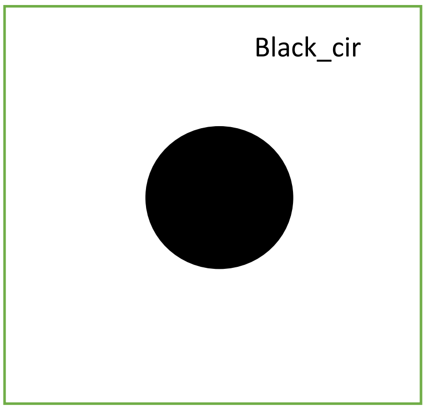
(ii) Rotate image by 45degrees and plot the Fourier transform. [5]

(iii) Mutiply the output of part(i) with image an white_cir.jpg image and generate the inverse fourier transform output image. Write down your observations in detail. [10]

(iv) Multiply the output of part(i) with image an black_cir.jpg image and generate the inverse fourier transform output image. Write down your observations in detail. [10]







4. DCT[20]

Perform DCT on image using your own Matlab function.

Input parameters:

1) image (512x512 matrix)
2) block_size (typically the block size is a power of two, in the example explained above, we were using an 8x8 block)
3) coef_number (this is the number of the coefficients that we are going to keep for each block, in the example explained above the number of coefficients we are keeping for each block is 10)

Output parameters:

1) decompressed (the decompressed image which is the output after taking the 2D IDCT)
2) map (an index map that indicates the locations of coefficients to kept in each block. This output is a binary matrix of size "block_size x block_size", which consists of values 0 and 1

only. If map(i,j) = 1 at a certain location, this means that we are keeping all the coefficients at the (i,j)th location in each block. If map(i,j) = 0, this means that we are discarding the (i,j)th coefficient in every block) .
3) energy (a matrix which gives the average energies at each location of the block. The way to compute the average energy for each location is explained above. This matrix should also be of size "block_size x block_size").
4) MSE/pixel

Plot the contents of the matrix "energy" (in 2-D). Do the described experiments for block_sizes 8 and 16. You may want to try different values for the number of coefficients to keep. Give the average block energy matrix, the block coefficient mask matrix, and the resulting MSE per pixel for the following cases in your report :
1) block size = 8, 10 coefficients kept 2) block size = 8, 5 coefficients kept 3) block size = 16, 40 coefficients kept 4) block size = 16, 10 coefficients kept.