EE 371R - DIGITAL IMAGE PROCESSING

PROF. AL BOVIK **UNIQUE NO.16705**

HOMEWORK #4

Due on Tuesday, November 14th, 2017

- The homework is due at the beginning of the class.
- You are encouraged to work with other students to understand the course material. However, sharing source code, images, or other answers from an outstanding homework is strictly forbidden. You are to submit your own work.
- Please include your written answer and/or output images for each problem.
- Please show all steps for full credit, and attach the code to each problem.
- Images needed for this homework can be downloaded from Blackboard.
- This homework will familiarize you with MATLAB. Use help and lookfor in MATLAB to learn more about suggested functions and commands.
- Traditional for/while loops are very inefficient in MATLAB. To encourage the development of vectorized (operating on arrays) code, unnecessary use of for/while (for example, to index each element of an image) will be penalized. However, tiny loops (say i = 1:256 or so) of vectorized instructions can be used.

1. Block Truncation Coding (BTC)

BTC is a fast, lossy compression technique. This problem will help you understand how BTC is used, along with some of its advantages and disadvantages. Note that 4x4 blocks are used where applicable in the following sub-problems. (Useful functions: ***col2im, im2col, bi2de, de2bi***.)

Figure 1. itjoker.jpg

(a) Write a function, ***mybtcmeanstd***, with the signature, ***function [mu , sigma] = mybtcmeanstd (img block , B1 , B2)***, to compute the mean (mu) and standard deviation (sigma) of an image block with B1 bits to store mu and B2 bits to store sigma.

(b) Write a function, ***mybtcthreshold***, with the signature, ***function [binary block] = mybtcthreshold (img block , mu)***, to compute the BTC binary block by thresholding an image block at the mean (mu).

(c) Write a function, ***mybtcdecode***, with the signature, ***function [decoded block] = mybtcdecode (binary block , mu , sigma)***, to decode a BTC binary block given the mean (mu) and standard deviation (sigma).

(d) Read in the image, ***itjoker.jpg***, and convert it into gray scale. Compute the block mean and standard deviation of the gray-scale joker.jpg using three different settings of B1 and B2: {B1, B2} = {2, 1}, {3, 4}, and {6, 5}. Show the original and the three block mean images in a 2x2 grid, and comment on what you observe.

(e) Compute the BTC binary blocks using the same three settings of B1 and B2 above. Show the original and the three binary block images in a 2x2 grid, and comment on what you observe.

(f) Decode the three BTC binary block images. Show the original and the three decoded images in a 2x2 grid, and comment on what you observe. Compute and report the compression ratios for the three BTC coded images.

2. Discrete Cosine Transform (DCT) and JPEG
The goal of this problem is to help you better understand the discrete cosine transform (DCT) and the principles of the JPEG baseline algorithm. Note that 8x8 blocks are used where applicable in the following sub-problems (Helpful functions: ***dct2, idct2, blockproc***).

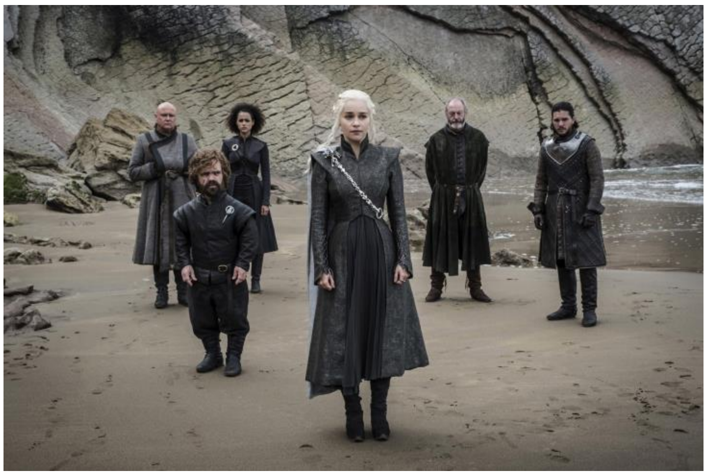

Figure 2. got.jpg

(a) Read in ***got.jpg*** and convert it to gray-scale. Perform the DCT on each non-overlapping block of size 8x8 in the gray-scale image. Show the original and DCT images side-by-side.

(b) Uniformly quantize each non-overlapping 8x8 block with the quantization matrix Q shown in slide 77 of Module 6. Compute the approximate compression ratio as the number of non-zero elements divided by the total number of pixels. Multiply the quantized image with the same quantization matrix to obtain the lossy DCT image. Show the lossy DCT image and report the approximate compression ratio.

(c) Perform the inverse DCT on each non-overlapping 8 x 8 block of the lossy DCT image. Compute the absolute difference between the original and reconstructed images. Show the reconstructed and absolute difference images side-by-side

3. Structural Similarity (SSIM) Index

The goal of this problem is to help you understand the structural similarity (SSIM) index and its relevance to subjective image quality and human perception. Note that 8 x 8 blocks are used where applicable in the following sub-questions (Helpful functions: *imnoise*, *padarray*, *im2col*, *mean*, *std*, *reshape*, *imagesc*).

(a) Read in the image ***liberty.jpg*** and convert it to gray-scale. Add Gaussian white noise of zero mean and σ=0.1 to the gray-scale image. Show the original and noisy images side-by-side with labels.

(b) Compute the block mean and standard deviation within each 8 x 8 neighborhood in the image. Show the block mean and standard deviation images of the original and noisy images side-by-side with labels.

(c) Compute the luminance similarity map ($L_{i,}$), contrast similarity map ($C_{i,}$), and structural similarity map ($S_{i,}$) between the original and noise images, where the stabilizing constants C_1=$(0.01\ x\ 255)_2$, C_2=$(0.03\ x\ 255)_2$, and C_3=C_2 / 2 . Combine the three maps to compute the SSIM index map. Show all four maps in a 2 x 2 grid with labels using *imagesc*. Compute and report the mean SSIM (SSIM index). Explain which aspect of the human visual system is explained by the SSIM map and index.

(d) Compute the SSIM map and SSIM index for five different Gaussian white noise levels of the gray-scale image: σ= 0, 0.01, 0.1, 0.3, and 1. Compare Peak Signal-to-Noise Ratio (PSNR) and SSIM index for each noisy image and report the relevance of perceptual image quality based on your observations. Display the noisy images and SSIM maps in a 5 x 2 grid with labels including σ, PSNR, and SSIM index. Plot PSNR and SSIM index against σ.

4. Image Quality

You are going to capture natural images using any current mobile device (e.g., a smartphone or tablet), rate their visual quality, and then assess them using a no-reference visual quality algorithm (BRISQUE).

You should take photos that you think are interesting and that you are willing to share. You will collect 8 photos in 6 different categories for a total of 48 photos. Half of the photos in each category should be indoor scenes and the other half outdoor scenes. At most, you may have 1 photo in each category predominately featuring a human face or faces. The categories are grainy (low-light noise), shaky (motion blur), blurry (out of focus), under-exposed, over-exposed, and no apparent visual distortions (see example images from each category below). Many photos will feature multiple types of distortion, but please categorize them based on what you believe to be the most dominant distortion.

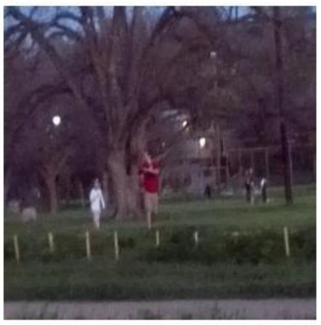
Figure 3. Grainy

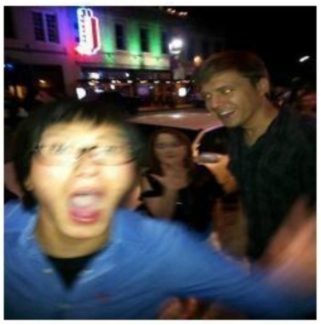
Figure 4. Shaky

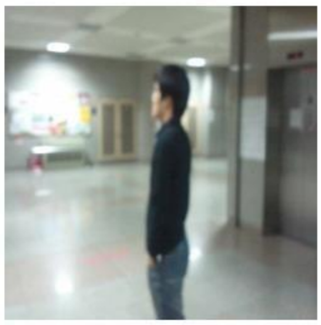
Figure 5. Blurry

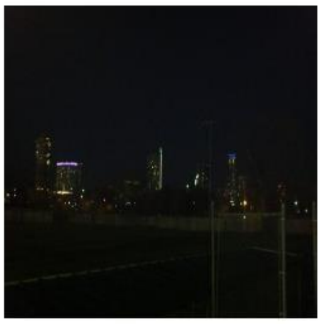
Figure 6. Under-exposed

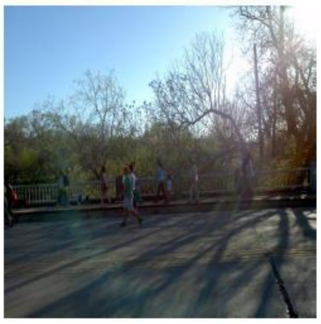
Figure 7. Over-exposed

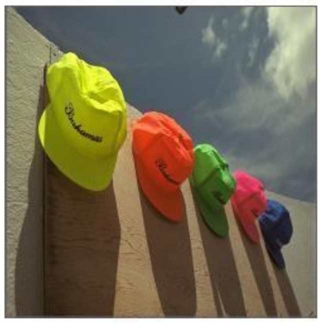
Figure 8. No apparent distortion

Dos

Pictures should be captured under varying illumination conditions (i.e., captured in both day and night lighting conditions).

Pictures should contain interesting content.

If your pictures feature people prominently, please ensure that they have no objection to the viewing or use of their pictures by the class or by others as they may be part of a public dataset in the future.

Don’ts

Try not to take uninteresting pictures like these:

Example-1

Example-2

Example-3

Example-4

Do not submitted processed pictures, i.e., pictures should not be processed by any aesthetic filters provided by applications such as Instagram or flickr.

The pictures should not be downloaded from the web.

The pictures should not contain duplicate or similar image content.

(a) Name your images in the following format <img_number> where img_number is just a numerical index. Please zip these images and submit them on Canvas along with your homework.

(b) Create an excel file to store information about each image. In the first column, please provide information describing the device that you used to capture each image, e.g., iPhone 5/6, Samsung Galaxy III, HTC One, etc. The row should correspond to the associated image number.

(c) Judge the quality of each of your images on a scale of 1-5 (poor, bad, fair, good, and excellent) when viewing them on the same device on which you captured them. Record these scores in the second column of the excel file.

(d) You will now use a trained BRISQUE model to predict image quality scores. Unzip BRISQUE_release.zip (available on Canvas) and read readme.txt for help using the available MATLAB functions. Display ***testimage1.bmp*** and ***testimage2.bmp*** (contained in BRISQUE_release.zip) side-by-side and report their objective quality scores using BRISQUE.

(e) Apply BRISQUE to your 48 images and record the scores in the third column of the excel file. Please submit this excel file on Canvas along with your homework.

(f) Report the Pearson’s linear correlation coefficient (LCC) and Spearman’s rank ordered correlation coefficient (SROCC) between the quality scores you assigned to the pictures in part (c) and the scores predicted by BRISQUE in part (e) (helpful function: ***corr***).