Assignment 3

General Info

- The work can be done anywhere where MATLAB and the related toolboxes are available.
- A **written report** is required. The report is free form but should include results, figures, and any code you wrote, as well as discussions of what you observe.
- Reports should be submitted as an electronic copy in **PDF** format on **Canvas** before the due date.
- Late submissions will not be accepted.
- Better documentation and clearer discussions of your work improves our ability to fairly mark your report. Make sure your report is well structured, organized, and clear. Your report has to follow the order of questions. Give all relevant code right before the results and discussion of each part, not in an appendix.
- This assignment has 6 questions.

1. Redundancy in the Color Space

Here you will explore how one can account for information that is perceptually redundant, specifically, visually redundant color information.

The YC_BC_R colour space is derived from the RGB colour space and has the following three components: Y, Luminance or Luma component obtained from RGB after gamma correction; C_B=B-Y, how far the blue component is from Luma; and C_R=R-Y, how far the red component is from Luma. This colour space separates the luminance and chrominance components into different channels, and is mostly used in compression for TV transmission.

(a) Load ‘peppers.png’ (already included in default MATLAB path) into MATLAB with *imread*. Convert the image into YC_BC_R and spatially downsample the luma channel Y by a factor of 4 using a ‘bilinear’ interpolator. Upsample the downsampled luma channel to its original size, then compute and report the mean squared error with the original luma channel. Next, reconstruct the image by combining the resampled luma with the original C_B, and C_R information and converting it back to RGB. Compare this reconstruction with the original ‘peppers.png’ and comment on your observations.

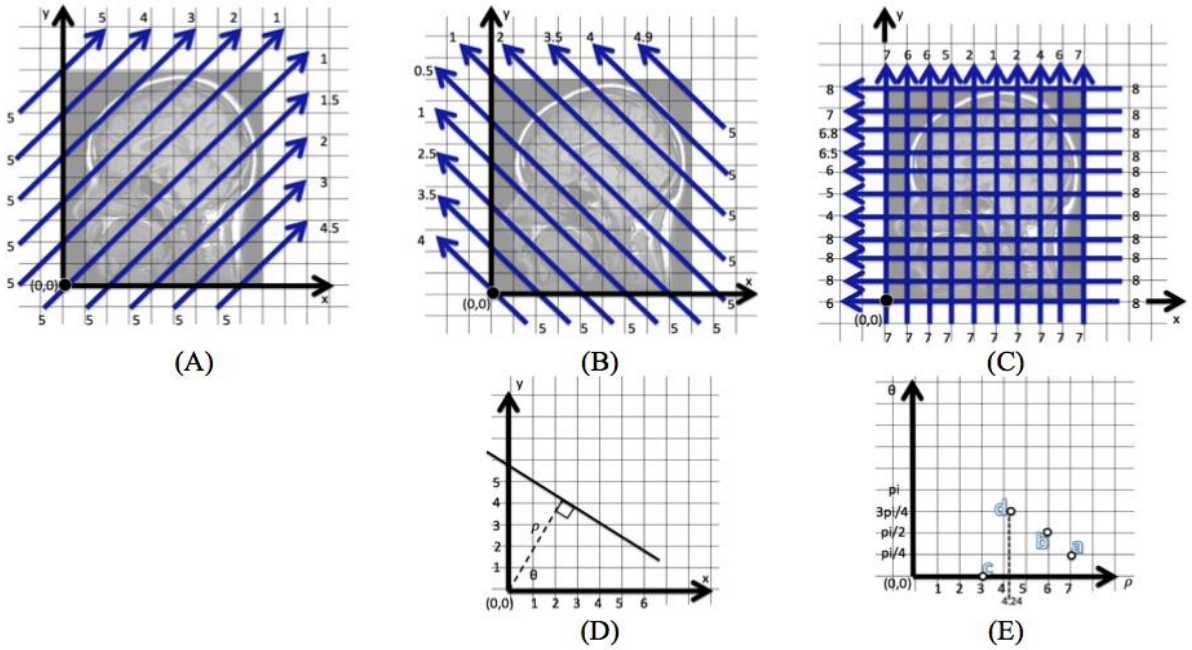
(b) This time, rather than resampling the luma, downsample **both** chroma channels C_B, and C_R. Measure and report the mean squared error between each resampled channel and its original values separately. Finally, reconstruct the image by combining the two resampled chroma channels with the original luma information and converting it back to RGB. Compare this reconstruction with the original ‘peppers.png’ and comment on your observations.

(c) Which of the two iterations resulted in a better looking image? Why? How is this relevant in the context of image compression?

2. Reconstruction from Projections & CT

Figures (A-C) represent X-ray beams passing through a slice of brain tissue. The numbers at the tail and head of each arrow indicate the intensities of the X-ray beam at the source and detector, respectively. The grid lines are represented by 1 unit. The relationship between the signal strength at the transmitter I_0 and signal strength at the receiver I is defined by the following equation:

$$I = I_0 \exp\left(-\int \mu(x,y)ds\right)$$



(a) What is the relationship between the input and the output signal strength and the radon transform? Consider the equation given above for understanding this relationship.

(b) Figure (D) shows how a line in x-y space is represented using ρ and θ. Your goal is to report the Radon transform values at the 4 indicated points (a-d) in Figure (E), e.g. point d is at (ρ, θ) = $\left(3\sqrt{2}, 3\pi/4\right)$

Now, you will explore different interpolators and filters that can be employed to enhance the 2D reconstruction from linear projections. You will use the Signal-to-Noise Ratio and Mean Squared Error similarity metrics to evaluate the effectiveness of each parameter. The `iradon` function in MATLAB automatically includes the following filters and interpolators which you will be investigating.

```
interpolators = {'nearest';'linear';'spline';'pchip';'cubic';'v5cubic'};
filters = {'none';'Ram-Lak';'Shepp-Logan';'Cosine';'Hamming';'Hann'};
```

Load the Q2_student.m file and perform the following:

(c) Try all pairwise-combinations of the parameters listed above. Which pair of parameters produces the best results in terms of a low MSE and a high SNR? (**Hint:** use loops to

automate the process!) Does this correspond with what you visually observe to be the best?

(d) Change the number of angles from 200 to 50 with steps of 50 degrees. Are the parameters chosen for part (c) still the best candidates? Why or why not?

3. JPEG Compression

JPEG compression is a commonly used method of compression for digital images. The degree of compression can be adjusted, allowing a selectable trade-off between storage size and image quality.

The JPEG encoding process in color images comprises the following 5 steps:

1. The representation of the colors in the image is converted from RGB to YC_BC_R.
2. The resolution of the chroma data is reduced, usually by a factor of 2.
3. The image is split into blocks of 8×8 pixels, and for each block, each of the Y, C_B, and C_R data undergoes the Discrete Cosine Transform (DCT). A DCT is similar to a Fourier transform in the sense that it produces a kind of frequency spectrum.
4. The amplitudes of the frequency components are quantized.
5. The resulting data for all 8×8 blocks is further compressed with a lossless algorithm, a variant of Huffman encoding.

In question 1 (“Redundancy in the Color Space”), you observed the effects of subsampling the color space (steps 1 and 2 above). In this problem you will focus on steps 3 and 4. For the sake of simplicity, you will perform JPEG compression on a scalar gray scale image from the default MATLAB path: *`cameraman.tif`*.

(a) Use MATLAB’s `dctmtx` function to generate an 8x8 DCT transformation matrix. Plot this matrix using `imshow` and briefly explain what the matrix does.

(b) Prior to performing DCT on the image *cameraman.tif*, the intensity values must be shifted from a positive range of [0,255] to one centered around zero [-128,127] by subtracting 128 from every intensity value. Explain why.

(c) MATLAB’s documentation for `dctmtx` states: “In JPEG compression, the DCT of each 8-by-8 block is computed. To perform this computation, use `dctmtx` to determine D, and then calculate each DCT using D*A*D' (where, A is each 8-by-8 block).” To perform DCT on every 8x8 block of the image, use MATLAB’s `blockproc` function. Plot the resulting transformed image and explain the results.

(d) To quantize the transformed image, the following matrix is used:

```
Q =
     1     1     1     2     2     2     4     4
     1     1     2     2     2     4     4     4
     1     2     2     2     4     4     4     8
     2     2     2     4     4     4     8     8
     2     2     4     4     4     8     8     8
     2     4     4     4     8     8     8    16
     4     4     4     8     8     8    16    16
     4     4     8     8     8    16    16    16
```

And the quantized DCT coefficients are computed as:

```
B = round(G./(q_level*Q)),
```

where, *G* is the un-quantized DCT coefficients, *Q* is the quantization matrix, *q_level* is the level of quantization we want to achieve, and *B* is the resulting quantized (normalized) coefficients. To achieve a higher level of quantization, *G* is divided by integer multiples of *Q*. You will be asked to perform this quantization in the next question. For this part, just briefly explain why this quantization matrix has the given values.

4. JPEG Decoding

You will now explore step 4 of JPEG compression. Before performing data compression, you will study the effects of quantization of DCT coefficients.

(a) Perform the quantization as described in Q3(d). As you divide each block by the quantization matrix and round it, you are increasing the sparsity of how the image is represented effectively reducing many values of *G* to zero. To observe this, plot the histogram of *G* using the `hist` function and compare it to the histogram of *B* with *q_level*=10. Briefly comment on the histograms.

(b) Now that you quantized and compressed the image, try to reconstruct it. The first step is to shift back the coefficients. To reconstruct, multiply *B* by *Q* the same number of times that you divided it in the quantization step to obtain a new *G*:

```
G = B.*(q_level*Q)
```

Now perform the inverse block-wise DCT. The forward DCT transform was defined as G=D*A*D', the inverse DCT transform is defined as A=D'*G*D. Plot the reconstructed image as shown below and explain the differences compared to the original image.

Hint: Remember to add back 128 and round the image values to [0 255].

(c) Perform JPEG compression at different levels of quantization and explain your observations. What happens at very high q_level values? Zoom in on the camera region of the image and explain the artefacts that you observe as a result of this quantization.

5. Lossless (Huffman) Encoding

In this problem, you will focus on step 5 of the JPEG encoding process. The JPEG compression standard uses two forms of lossless compression, run-length and Huffman, to further compress the quantized DCT coefficients. Explore the effects of Huffman coding on the cameraman image:

(a) Briefly explain information entropy and its relationship to compression.

(b) Compute the entropy of the image and calculate the maximum compression that can be expected.

(c) Check whether the maximum compression calculated in 5(b) is obtained for your data, or if not, how close it gets to that.

6. Predictive Coding

Predictive coding is a lossless compression technique that allows one to further compress an image. A simple way to employ predictive coding on an image is by storing the difference between adjacent rows in an image rather than the pixel values such that:

$$e(x,y) = f(x,y) - \hat{f}(x,y)$$

$$\hat{f}(x,y) = \text{round}[\alpha f(x,y-1)], \alpha = 1.$$

(a) Encode the *cameraman.tif* image using this predictive scheme and plot the resulting (predicted) image. What does this image look like?

(b) To appreciate why this simple method can produce good compression, plot the histogram of the cameraman image before and after the predictive encoding step and comment on the differences between the two. Calculate the entropy of the two histograms to quantify the difference between potential compression ratios that may be achieved.

(c) Reconstruct the predictive encoded image and calculate the error between the original and reconstructed versions. Is this method truly lossless, why or why not?

End of assignment 3