DIP Course

MiniProject 3

Vector Quantization and Image Compression

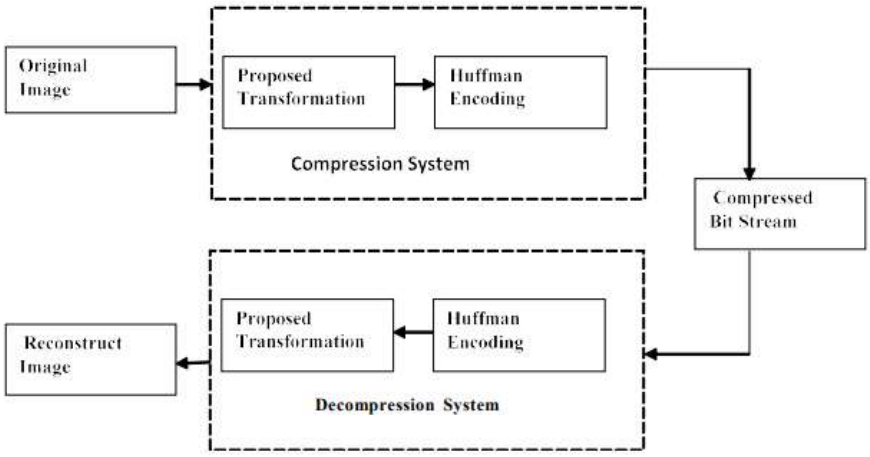
Part1: Color-Based Segmentation Using K-Means Clustering

Clustering is a way to separate groups of objects. K-means clustering treats each object as having a location in space. It finds partitions such that objects within each cluster are as close to each other as possible, and as far from objects in other clusters as possible.

1. Read in yellowlily.jpg which is an image of three particular parts (flower, flower leaves, soil)
2. Classify the Colors in RGB Space Using K-Means Clustering and label Every Pixel in the Image Using the Results from k-means.
3. Create Images that Segment flower (yellow part of image) by Color.

Part 2: Lossless Compression

In this part we will use a lossless compression technique called Differential coding to compress source image and then decompress it at destination.



A) Encoder

It mainly consists of two blocks; a proposed transformation and a Huffman Encoder.

1. Use ‘lenna.tif’ as the input image, convert it to grayscale and plot its histogram.

2. Compute the entropy of the grayscale image by using the entropy function in Matlab.

3. Use the lossless predictive coding algorithm (differential coding) and create the prediction error image.

Use these equations for differential lossless coding:

$$e(m, n) = f(m, n) - \hat{f}(m, n)$$

$$\hat{f}(m, n) = \begin{cases} 0, & if\ \ m = 1\ or\ \ n = 1 \\ \mathrm{f(m, n-1)}, & o.w \end{cases}$$

4. Show the prediction error images and plot its histogram.

5. Compare the histogram of the input image and prediction error image and explain their difference.

6. Write a function to compute the entropy of a random variable x and probabilities for the occurrence of each symbol then Compute the entropy for prediction error image.

$$function[H, symbol, p] \ = \ entr(x)$$

7. Encode the prediction error images by using Huffman coding. (use these Matlab functions: "huffmandict" & "huffmanenco")

8. What is the average code word length per symbol?

B) decoder

It mainly consists of two blocks; proposed transformation and Huffman Decoder.

9. Create prediction error image in destination. (Hint: use Huffman decoding Matlab function "huffmandecode")

10. Decode prediction error image and show decompressed image then compare with original input. What is your conclusion?

Part 3: Lossy compression

A) KLT

Use 'lenna.tif' as the input image and compute its KL-transform. To do this, consider each vector in RGB space as a random variable vector x and compute the covariance matrix of these vectors (n=3).

$$m_x = E\{x\} \qquad x: n \times 1$$

$$C_x = E\{(x - m_x)(x - m_x)^T\} \quad C_x: n \times n$$

Use the eigenvectors of this matrix to obtain the KLT of the image.

1. Show separately each component of RGB and KLT as a grayscale image (attention for normalizing the values before showing the images). Comment on what you observe from images.

2. In KLT space, replace respectively one and two components by zero and compute inverse KLT to obtain again the image in RGB space (the components correspond to the smaller eigenvalues). Show the resulting images and compute MSE between these images and original RGB images. Comment on the results.

B) DCT

1. Use ‘cameraman.tif’ as the input image.

2. Show the input image and the absolute fft of input image.

3. Compute the two-dimensional DCT of the non-overlapping sub-images of 8x8 on ‘cameraman.tif’ image.

4.What are the indices of the largest DCT coefficients? Using DFT, show the 2D power spectrum of resulting images.

5.What happens if you set 50% of lowest frequency coefficients to zero and perform the inverse DCT? What image information is contained in the low-frequency coefficients? What is contained in the high-frequency coefficients?