Data Compression Homework #4
Due Date: November 27, 2012

NAME: Shiyun Liu
Number: G45472436

Project:

In this assignment, you will implement a few parts of a simplified Baseline JPEG. Specifically, you will implement the quantizer/dequantizer, and use the built-in dct transform and its inverse. Although for the calculations requested below you need to know the size of the coded bitstreams, you should note that the actual coded bitsreams need not be computed, and therefore, you should not implement the entropy coder/decoder stage of JPEG.

Your implementation should take as input the original image and a scalar for the quantization matrix. Conceptually, the output of the compression is a bit stream consisting of a header and the data; the header contains information that specifies the two Huffman tables (one for the DC terms, and one for the AC terms); the data consist of two parts: the first part is the bit stream for the DC terms, and the second part is the bit stream for the AC terms. Note that your implementation is for gray-scale images only, and that you should use one specific quantization matrix (the one given in the lecture notes). In particular, since the quantization matrix is fixed, you need not store it in the output of the coder.

To simplify matters, we remove the restriction that the Huffman codewords must be at most 16 bits long. Rather, you can allow them to be of any length.

a) Apply your algorithm on the grayscale version of the River image (that you usd in Homework 3), using different scalars(for multiplying the quatization matrix), and identify which scalars give compression ratios 12, 15, and 20.

b) For each compression ratio indicated above, compute the SNR.

c) Plot the compression ratios indicated above against the corresponding scalars (the x-axis is the scalars, the y-axis is the compression ratios). Also, plot the SNR's against the same scalars.

d) Repeat a-c on another image of your choice. Make sure you show the original image.

Answer:

a)

When scalar ≈ 1.5 , CR = 12.

When scalar ≈ 2.0 , CR = 15.

When scalar ≈ 2.9 , CR = 20.

b)

When CR = 12 , SNR = 22.4964.

When CR = 15 , SNR =21.7230.

When CR = 20 , SNR =20.7105.

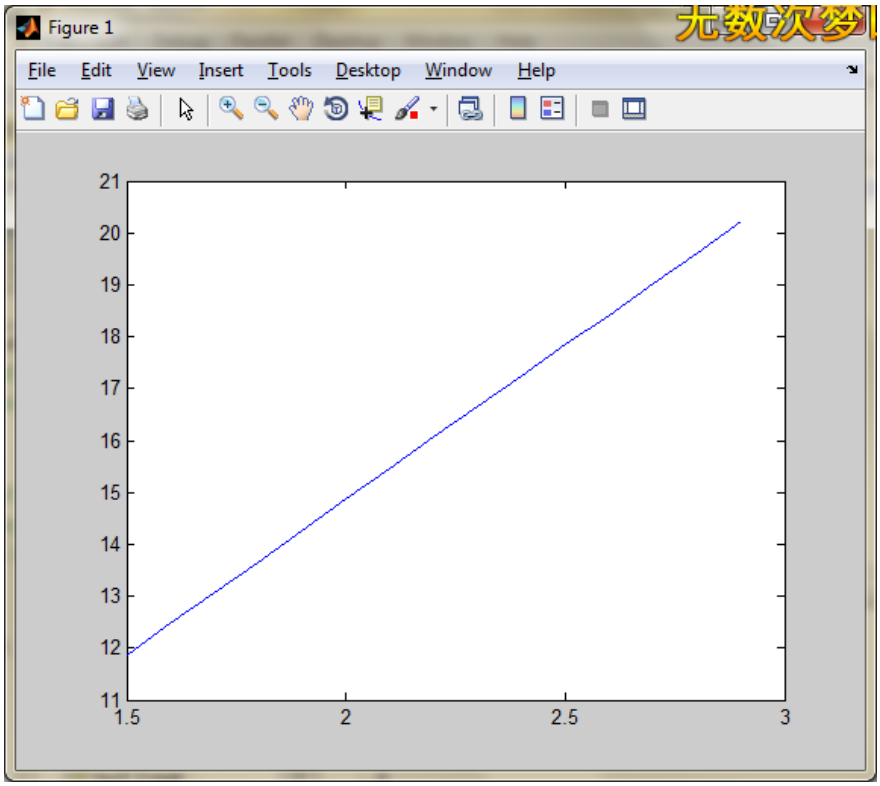
c)

I use 0.1 as the difference between two points of scalar. In this work, the smallest scalar is 1.5 and the largest scalar is 2.9, so I choose 1.5,1.6,1.7……2.8,2.9 as the x-axis value of the points.

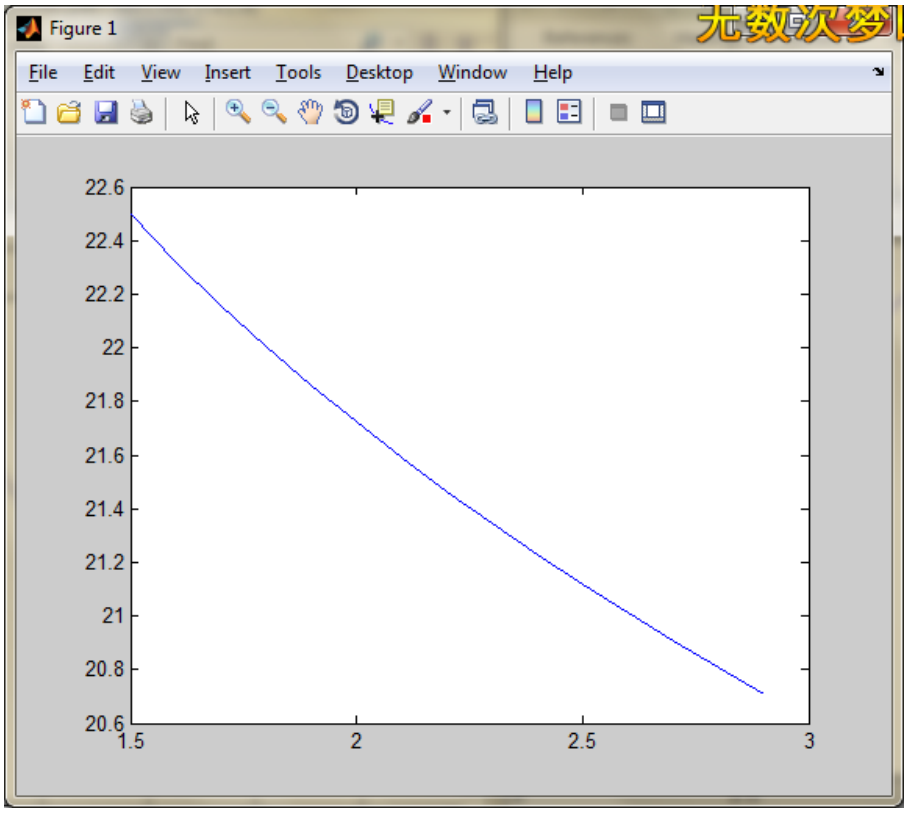
The first figure has the corresponding compression ratios as the y-axis value of the points. And the second figure has the corresponding SNR as the y-axis value of the points.

Both of these two figures depend on these 15 points.

The figure whose x-axis is the scalars and the y-axis is the compression ratios :

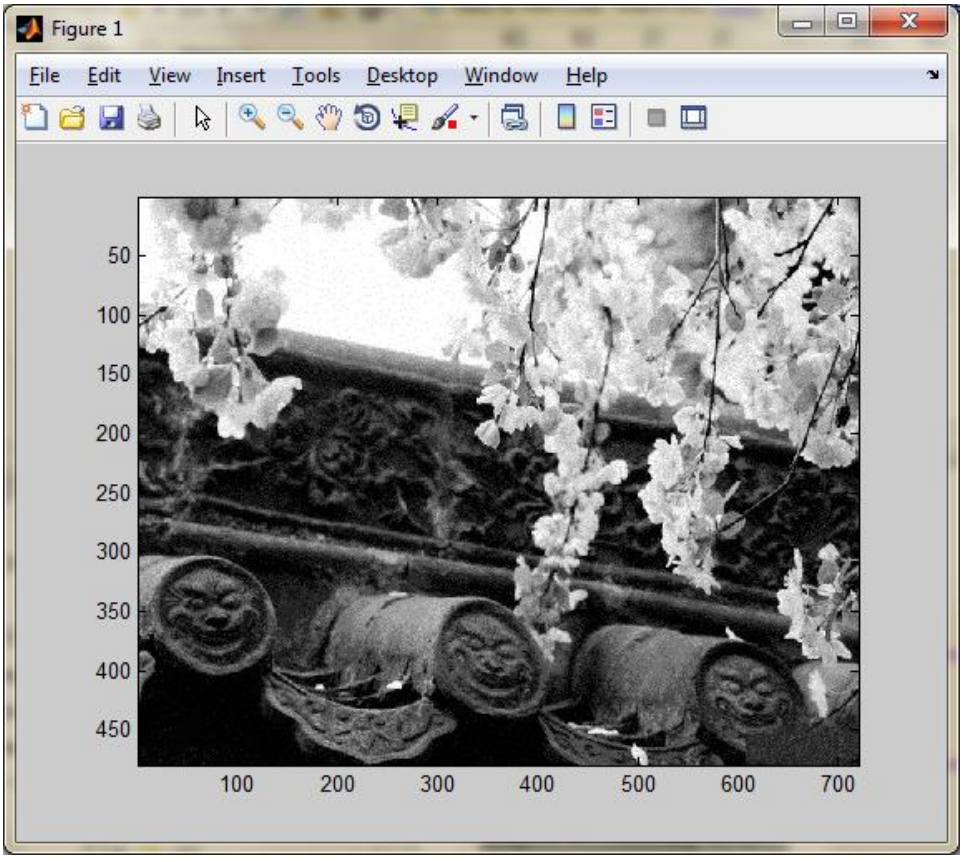


The figure whose x-axis is the scalars and the y-axis is the SNR :



d) My Choose “flower.gif” (480*720) as my original picture:

After changing into grey picture, this picture becomes:



(1) When scalar ≈ 2.1 , CR = 12.

When scalar ≈ 2.7 , CR = 15.

When scalar ≈ 3.9 , CR = 20.

(2) When CR = 12 , SNR = 18.9821.

When CR = 15 , SNR = 18.6710.

When CR = 20 , SNR = 18.1958.

(3) I also use 0.1 as the difference between two points of scalar. In this work, the smallest scalar is 2,1 and the largest scalar is 3.9, so I choose 2.1, 2.2, 2.3, 2,4 ……3.7, 3.8, 3.9 as the x-axis value of the points.

The first figure has the corresponding compression ratios as the y-axis value of the points. And the second figure has the corresponding SNR as the y-axis value of the points.

Both of these two figures depend on these 15 points.

Figure whose x-axis is the scalars and the y-axis is the compression ratios :

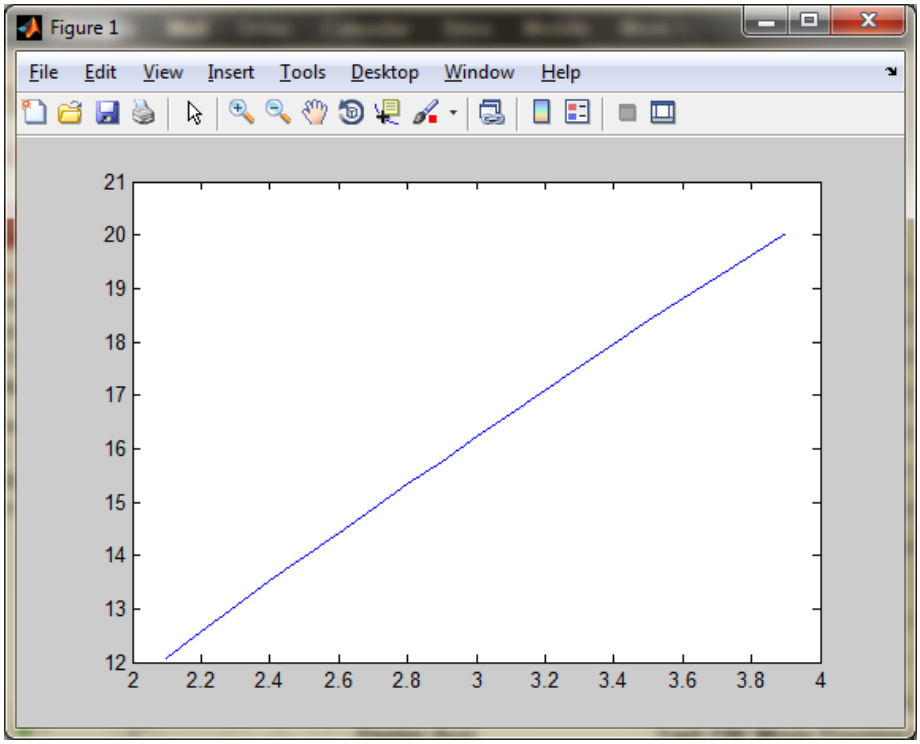


Figure whose x-axis is the scalars and the y-axis is the SNR :