1. JPEG Compression

In this homework, we request that you can achieve the JPEG compression using MATLAB or other platforms. You can use the library functions to perform some operations such as color transform or DCT. But the truncated Huffman coding part must be done by your own codes. The whole processes are shown as following:

(1) Color transform

The input “Lena” image is a colorful one in RGB space. According to the request of JPEG, you need to change it to YCbCr space (rgb2ycbcr in MATLAB). To simplify the process, you just need to keep the Y channel which is actually the intensity. You will not use the other two Cb and Cr in this homework.

(2) Image segmentation

The intensity image (Y channel) will first be segmented into 8*8 blocks. For example, the size of the “Lena” image is 512*512, so there will be 4096 different blocks with the size of 8*8 in total.

(3) DCT

Implement the DCT for each 8*8 block. (dct2 in MATLAB)

(4) Quantization

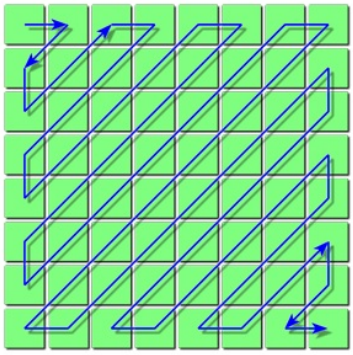
To quantize the DCT results, you just need to calculate the blocks divided by a quantization table. For the Y channel, the official table is shown in Fig. 1. The table has the same size 8*8.

```
Y_Table=[ 16  11  10  16  24  40  51  61 ; ...
          12  12  14  19  26  58  60  55 ; ...
          14  13  16  24  40  57  69  56 ; ...
          14  17  22  29  51  87  80  62 ; ...
          18  22  37  56  68 109 103  77 ; ...
          24  35  55  64  81 104 113  92 ; ...
          49  64  78  87 103 121 120 101 ; ...
          72  92  95  98 112 100 103  99 ];
```

Figure 1: Quantization table for the Y channel.

(4) Zig-Zag scan

Before the next coding step, you need to change the 8*8 blocks to a 1*64 vector using the Zig-Zag scanning method (see Fig. 2). It is quite simple to realize and if you still don’t know how to do, try to check some references.



```
zigzag = [ 0,  1,  8,  16,  9,  2,  3,  10, ...
           17,  24,  32,  25,  18,  11,  4,  5,  ...
           12,  19,  26,  33,  40,  48,  41,  34,  ...
           27,  20,  13,  6,  7,  14,  21,  28,  ...
           35,  42,  49,  56,  57,  50,  43,  36,  ...
           29,  22,  15,  23,  30,  37,  44,  51,  ...
           58,  59,  52,  45,  38,  31,  39,  46,  ...
           53,  60,  61,  54,  47,  55,  62,  63];
zigzag = zigzag + 1;
```

Figure 2: Zig-Zag scan.

(4) Coding

In this step, we request that you use the truncated Huffman coding method to deal with the Zig-Zag scanning result (1*64). Please note that it is the truncated one not the original Huffman coding method. Try to read the courseware if you meet some difficulties and there are some reference functions for the original Huffman method in MATLAB (huffmandict, huffmanenco and huffmandeco).

(5) Reconstruction

In fact, you can take the reconstruction steps as the inverse processes of the above operations.

- Decode

Realize the decoding of the truncated Huffman coding method and get the decoding 1*64 vector from your Huffman coding results.

- Inverse Zig-Zag

Recover the 8*8 blocks from the decoding 1*64 vector by the inverse Zig-Zag scanning (see Fig. 3).

```
inzigzag=[0,   1,   5,   6,  14,  15,  27,  28,...
     2,   4,   7,  13,  16,  26,  29,  42,...
     3,   8,  12,  17,  25,  30,  41,  43,...
     9,  11,  18,  24,  31,  40,  44,  53,...
    10,  19,  23,  32,  39,  45,  52,  54,...
    20,  22,  33,  38,  46,  41,  55,  60,...
    21,  34,  37,  47,  50,  56,  59,  61,...
    35,  36,  48,  49,  57,  58,  62,  63];
inzigzag = inzigzag + 1;
```

Figure 3: Inverse Zig-Zag scan.

- Inverse quantization

Multiply each 8*8 block by the quantization table (see Fig. 1).

- IDCT

Perform the inverse discrete cosine transform. (idct2 in MATLAB)

- Image Reconstruction

Reconstruct the 512*512 image from your 4096 IDCT 8*8 results.

(6) Evaluation

We need you to calculate some parameters to evaluate the coding result.

- Root mean square error

$$e_{rms} = \left\{ \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left[\hat{f}(x,y) - f(x,y) \right]^2 \right\}^{1/2} \tag{1}$$

- Mean-square signal-to-noise ratio

$$SNR_{ms} = \frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left[f(x,y) \right]^2}{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left[\hat{f}(x,y) - f(x,y) \right]^2} \tag{2}$$

- Compression ratio

$$R = \frac{\#\,Bits \quad before}{\#\,Bits \quad after} \tag{3}$$

2. Results

We request that you write a description or plot a flowchart for your truncated Huffman coding method. Then, you need to display the Y channel of “Lena” images before and after the compression. Also, you need to give the truncated Huffman code of the given 8*8 block shown in Table 1, which is the last block of the first row from the input image. You can type the result directly or just take a screenshot of the matrix in your processing platform. Finally, you need to show the three evaluation parameters.

Table 1: A cropped 8*8 block from the “Lena” image.

134	156	160	162	163	162	149	126
134	156	160	162	163	162	149	126
134	156	160	162	163	162	149	126
134	156	160	162	163	162	149	126
134	156	160	162	163	162	149	126
134	139	144	141	135	122	104	82
125	126	121	112	95	84	71	66
114	97	87	76	74	59	63	56