4C08 Lab3 Report

Yiqian Zhang - 19320271

March, 2020

1 Variable Length Coding

Q1a. Calculate the Entropy of this quantised image

$$H(X) = -\sum_x p(x) log_2 p(x)$$
$$= -(\frac{5}{8}log_2(\frac{5}{8}) + \frac{3}{32}log_2(\frac{3}{32}) + \frac{3}{32}log_2(\frac{3}{32}) + \frac{1}{32}log_2(\frac{1}{32}) + \frac{1}{8}log_2(\frac{1}{8}) + \frac{1}{32}log_2(\frac{1}{32}))$$
$$= 1.75bit$$

Q1b. Using the Huffman Coding method, design a set of variable length codewords for this image. Include your Huffman Coding Tree in your report.

Variable length codewords is as shown in Table 1. Figure 1 presents Huffman coding tree of this image.

Level	Code
1	0
2	110
3	101
4	1111
5	100
6	1110

Table 1: Variable length codeword list

Q1c Calculate the average codeword length using your designed codewords.

The average code word length is 1.81bit (Equation 1), which is greater than the entropy. Entropy is the minimum average codeword length in theory, and the average codeword length produced by any coding method can only close to this value.

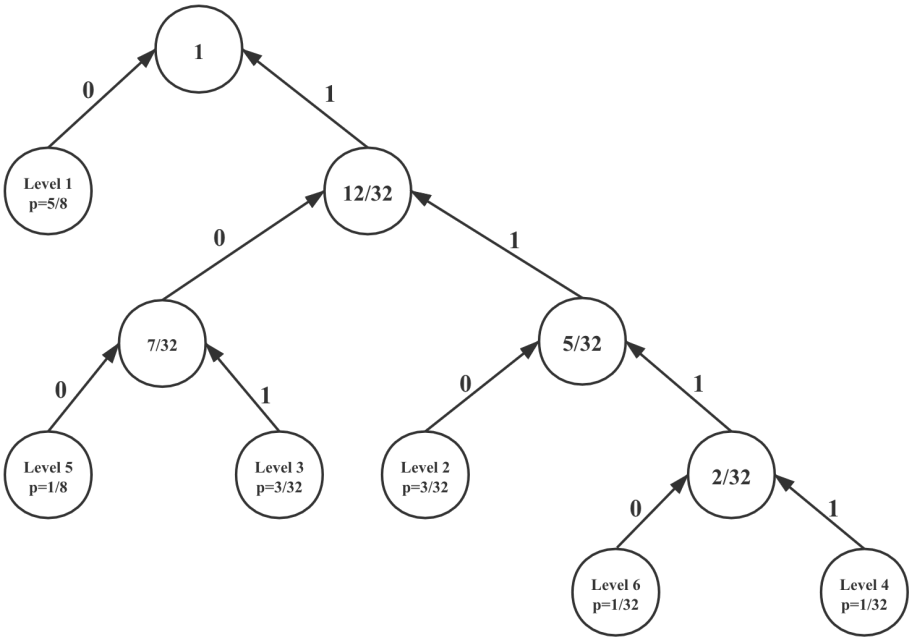

Figure 1: Huffman Coding Tree

$$H = \sum_{k=1}^{6} p_k l_k = \frac{5}{8} \times 1 + \frac{3}{32} \times 3 + \frac{3}{32} \times 3 + \frac{1}{32} \times 4 + \frac{1}{8} \times 3 + \frac{1}{32} \times 4 = 1.81 bit \quad (1)$$

2 Calculating Image Entropy

Q2a Write a MATLAB function calcEntropy that calculates the Entropy of an image.

```
function entropy = calcEntropy(img)
  %This function takes as input a 2D array Y containing
  %the image intensities and returns the entropy.

  edges = min(img(:)) : max(img(:));
  X = histcounts(img, edges, 'Normalization', 'probability');
  H = X .* log2(X);
  H(isnan(H)) = 0;
  entropy = -sum(H);

end
```


Figure 2: Left: Unquantised image; Right: Quantised image

Q2b Estimate the entropy

The entropy of the original is 7.0817bit.

Q2c Estimate the entropy of the quantised image using the given Q_{step} value.

```
function entropy = quantisedEntropy(img, Q_step)

  img = Q_step * round(img / Q_step);
  edges = min(img(:)) : max(img(:));
  X = histcounts(img, edges, 'Normalization', 'probability');
  H = X .* log2(X);
  H(isnan(H)) = 0;
  entropy = -sum(H);

end
```

Q2c Estimate the entropy of the quantised image using the given Q_{step} value.

Q_{step} is set to 50, and the entropy of the quantised image is 2.0071 bit.

Q2d Explain any difference these two entropies.

After quantisation, the image definition reduces. It will result in the decrease of entropy.

Figure 3: Haar transform; Left: The original Haar transform result; Right: optimised for visualisation

Q2e Write a MATLAB function to calculate the MSE between 2 images.

```
function MSE = calcMSE(Y1, Y2)
  % This function takes as input two 2D array Y1 and Y2 containing
  % the image intensities of two pictures and returns the mean ...
      square error
  % between both Y1 and Y2.

  MSE = sum((Y1 - Y2) .^ 2, 'all') / numel(Y1);

end
```

Q2f What is the MSE between the quantised and unquantised images?

The MSE is 53.6908.

Q2g Comment on the visual quality of the quantised and unquantised images.

Quantisation reduces the number of colors or grayscales used in an image. It will result in banding artifacts and reduction of definition. The comparison is shown in Figure 2.

3 The 2D Haar Transform

Q3a Write a MATLAB function that implements the 1-level Haar Transform and outputs a image of its subbands.

The result image is as shown in Figure 3. For visualisation, we can change the brightness and contrast: LoLo /2 and 128 is added to the LoHi, HiLo and HiHi bands.

```
function H = calcHaarLevel1(Y)
  % This function takes as input a 2D array Y containing
  % the image intensities of a picture and returns the 1-level
  % Haar Transform
  % N=2 Haar Transform is implemented

  % test for odd input
  s = size(Y);
  % column check
  if mod(s(1), 2) ~= 0
    Y = Y(1 :  end-1, :);
  end
  % row check
  if mod(s(2), 2) ~= 0
    Y = Y(:, 1 :  end-1);
  end
  Y = double(Y);

  % define Haar kernel
  T1 = 1 / 2 * [1 1; 1 1];
  T2 = 1 / 2 * [-1 1; -1 1];
  T3 = 1 / 2 * [-1 -1; 1 1];
  T4 = 1 / 2 * [1 -1; -1 1];

  % Haar transform using convolution (much faster)
  % Note that here must be valid padding meaning that there is no
  % zero-padding. This can make sure only pixels of the original
  % image is transformed
  LL = conv2(Y, T1, 'valid');
  HL = conv2(Y, T2, 'valid');
```

```
31    LH = conv2(Y, T3, 'valid');
32    HH = conv2(Y, T4, 'valid');
33
34    % we have to re-sample pixels because the default step size is 1 ...
          in CONV2,
35    % but we actually need the step size to be 2
36    LL = LL(1 : 2 :  end, 1 : 2 :  end);
37    HL = HL(1 : 2 :  end, 1 : 2 :  end);
38    LH = LH(1 : 2 :  end, 1 : 2 :  end);
39    HH = HH(1 : 2 :  end, 1 : 2 :  end);
40    H = [LL HL;LH HH];
41
42  end
```

Q3b Calculate the resulting Entropy H_{qhaar} of the transformed image after quantisation has been applied.

The entropies of LoLo, HiLo, LoHi, HiHi subband are $2.4108bit, 0.0857bit, 0.1118bit$ and $0.0113bit$, respectively, when quantisation step size is 50. Therefore, the entropy of transformed image is $0.6549bit$, which is the average value of four entropies of subbands.

Q3c Is $H_{qhaar} < H_{qi}$? Why / Why not

The results show that $H_{qhaar} < H_{qi}$. The reason is that the Lo-Lo filter is a 4-point average filter, and the Hi-Lo filter and Lo-Hi filter calculate average horizontal gradient and average vertical gradient, respectively. Also, the Hi-Hi filter can be seen as a highpass filter. Only limited information is reserved after transformation, and the entropy thus reduces.

Q3d Compress and reconstruct your image using stepsizes of $Q_{step/2}$, Q_{step} and $2 \times Q_{step}$ and comment on the differences.

The step size Q_{step} is set to 50, and the reconstruction results are as shown in Figure 4. We can notice that as the quantisation step size increases, the texture and details of the picture gradually disappear and are replaced by banding artifacts.

Q3e Comment on the relationship between the entropy and the objective quality metric for the 3 quantisation step sizes.

Table 2 is the objective quality metric. As the quantisation step size increases, MSE values increase and entropies drop compared to the original image. It turns out that the image quality becomes poor.

(a) Original image (b) Q_step=25

(c) Q_step=50 (d) Q_step=100

Figure 4: Haar transformation using different step sizes

	With Haar transformation			**Without Haar transformation**			
	$Q_{step/2}$	Q_{step}	$2 \times Q_{step}$	$Q_{step/2}$	Q_{step}	$2 \times Q_{step}$	Original image
Entropy	2.3704	1.8843	0.7468	2.8846	2.0071	1.2590	7.0817
MSE	60.4495	75.3122	90.2437	27.5297	53.6908	88.8035	0

Table 2: Objective quality metric of the 2D Haar transform

Q3f Using the same quantisation step sizes, calculate the MSE for the case where quantisation is performed on the image directly (ie. no haar transform is applied).

Quantisated images without Haar transformation shows better qualities than images that are transformed into the Haar domain with regard to MSE and entropy.

This is because Haar transform coefficients are numerically close to their neighborhood coefficients in Hi-Lo, Lo-Hi and Hi-Hi subbands. These coefficients become the same after quantisation, meaning that some information, such as horizontal gradient and vertical gradient, is missing. This causes the texture of the picture to disappear. On the other hand, quantisation applied to original image only results in the reduction of grayscale

(a) Level 1 (b) Level 2 (c) Level 3

(d) Level 4 (e) Level 5 (f) Original image

Figure 5: Haar transformation using different transform levels

levels and less information lost.

4 The multi-level 2D Haar Transform

Q4a Write a MATLAB function that implements the n-level Haar Transform and outputs a image of its subbands.

```
1  function H = calcHaar(Y, n)
2    % This function takes as input a 2D array Y containing
3    % the image intensities of a picture and returns the 1-level
4    % Haar Transform
5    % n is the number of levels used.
6
7    H = calcHaarLevel1(Y);
8
9    if n == 1
10     return
```

```
11    end
12
13    for i = 1 : n-1
14      % split L-L area of previous transform results
15      t_area = H(1 : (2 ^ -i) * end, 1 : (2 ^ -i) * end);
16      % do Level-1
17      L_N = calcHaarLevel1(t_area);
18      % replace original results
19      H(1 : (2 ^ -i) * end, 1 : (2 ^ -i) * end) = L_N;
20    end
21
22  end
```

Q4b Calculate the multi-level Haar Transform of your image and the reconstructed image for level numbers from 1 to 5 using your quantisation step size. Calculate the entropy of the quantised transformed image. Comment on the how the number of levels chosen affects the entropy and image quality.

The step size is set to 50, and the objective quality metric is shown in Table 3. The image becomes crisp as the number of levels increases (Figure 5), and the banding artifacts become not obvious. The entropies rise while MSE values drop. It turns out that increasing the transform level can improve the coding quality.

	L1	**L2**	**L3**	**L4**	**L5**	**Original image**
Entropy	2.8858	3.7520	4.6659	5.6201	6.5736	7.0817
MSE	30.3908	20.4326	16.8023	16.4318	16.3159	0

Table 3: Objective quality metric of the multi-level 2D Haar transform