Imaging Processing

Shida Sheng, Trinity College

03/10/2020

Problem 1&2

In an attempt to compress an image by coding the intensity of each pixel in that image, the intensity was first quantised to 6 levels. For that image, the values and probabilities of these levels are given in the table below

Level	Intensity Value	Probability
1	0	5/8
2	43	3/32
3	86	3/32
4	129	3/32
5	172	1/8
6	215	1/32

- Q1a Calculate the Entropy of this quantised image.

From the Lecture notes, The entropy equation is described as followed.

$$H(x) = -\sum_{x}^{n} p(x) log_2 p(x) \tag{1}$$

By using this equation, I calculate the entropy with different level quantisation.

Listing 1: Matlab Code.

```matlab
clc;
clear;close all;
pic = imread('girlface.png');
% pic = double(pic);
figure(1);
imshow(pic)
%% the pdf
pa = 0.6250 , pb = 0.125,pc =0.0938,
pd = 0.0313
% where
%% Use the equation of entropy
result = -pa*log2(pa) - pb*log2(pb) - 2 *pc*log2(pc)- 2 *pd*log2(pd);
```

The entropy result is 1.7522.

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

The table shows 6 different levels quantisations. By using the Huffman coding method, The level 1 can be described as 0, level 2 can be described as 10 , level 3 can be described as 110, level 4 can be described as 1110, level 5 can be described as 11110, level 6 can be described as 111110. The Huffman Coding Tree is shown as below.

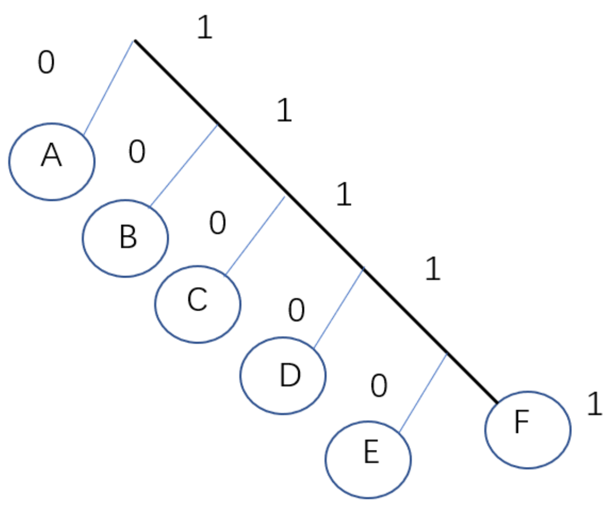

Figure 1: Huffman Coding Tree.

- Q1c Calculate the average codeword length using your designed codewords. Does it agree with the Entropy of the events in bits/pel. Explain any discrepancy.

This equation is to calculate the average of the Huffman Coding length.

$$Average.length = \sum p(k)l(k) \tag{2}$$

Listing 2: Matlab Code.

```
 Q_step = 43;

 pic1 = Q_step*round(pic/Q_step);
 pic2 = Q_step*round(pic/Q_step*2);
 pic3 = Q_step*round(pic/Q_step*3);
 pic4 = Q_step*round(pic/Q_step*4);
 pic5 = Q_step*round(pic/Q_step*5);
 figure(2);
 subplot(2,3,1),imshow(pic1), title('Level 2')
 subplot(2,3,2),imshow(pic2), title('Level 3')
 subplot(2,3,3),imshow(pic3), title('Level 4')
 subplot(2,3,4),imshow(pic4), title('Level 5')
 subplot(2,3,5),imshow(pic5), title('Level 6')
% calculate the entropy after quantisation
```

```matlab
entropy = calcEntropy(pic)
entropy1 = calcEntropy(pic1);
entropy2 = calcEntropy(pic2);
entropy3 = calcEntropy(pic3);
entropy4 = calcEntropy(pic4);
entropy5 = calcEntropy(pic5);
Allresult = [entropy,entropy1,entropy2,entropy3,entropy4,entropy5];
```

The result is [7.0817,2.1874,1.873,1.1863,1.1863,1.1863]. After doing quantisation to the image, the average length is much less than the original image. The reason is the image has been compressed, and it lost much details of the image.When the entropy is achieve its minimum, the entropy keep a same value.

Listing 3: Matlab Code.

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

% calculate histogram counts
p = imhist(Y(:));

% remove zero entries in p
p(p==0) = [];

% normalize p so that sum(p) is one.
p = p ./ numel(Y);

entropy = -sum(p.*log2(p));
end
```

This is the Function of calculate the entropy of image. First,finding out the distribution of all the pixels in the image and remove the zero entries in the distribution.After that, doing the normalization with the distribution and calculate the sum of the probaility times the pixels value.
Write a MATLAB function to calculate the MSE between 2 images. The function definition should be as below:

Listing 4: Sample Matlab code.

```matlab
function MSE1 = 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
% change to double
```

```

    Y1 = double(Y1);
    Y2 = double(Y2);

% follow the equation of Meam square error
% norm is to know abs difference between Y1 and Y2,
% numel is to get the total element number
err = (norm(Y1(:)-Y2(:),2).^2)/numel(Y1);
MSE1 = err;
end
```

The MSE equation can be described as

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y_i})^2 \tag{3}$$

After using this code, I calculate the MSE value with different quantisation level(2-6).
The result is [124.2] [7375.2] [15422.6] [16428.8] [17835.5]
With the quantisation level increasing, the MSE value becomes much greater. It means the image have a big different to the original image. The higher quantisation level, the more image details lost.
These are the different quantisation levels of the image.

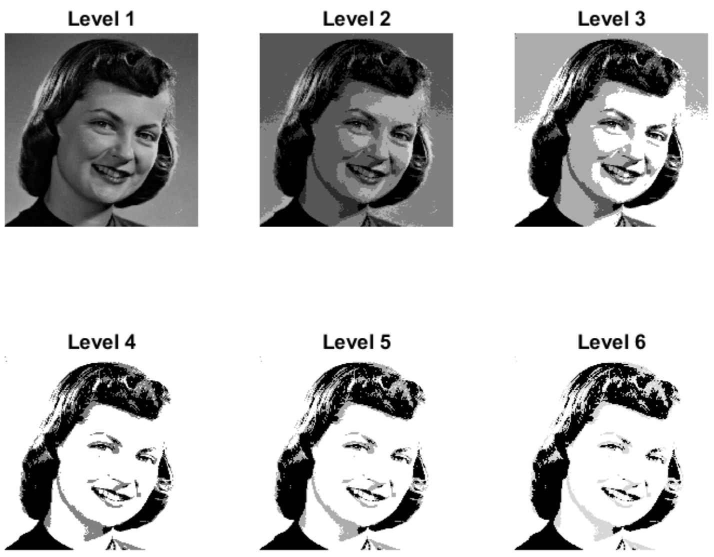

Figure 2: Huffman Coding Tree.

Problem 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 function definition should be as follows:

$$\text{If } \mathbf{x} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \text{ then } \mathbf{y} = \frac{1}{2}\begin{bmatrix} a+b+c+d & a-b+c-d \\ a-c+b-d & a-b-c+d \end{bmatrix}$$

Figure 3: Haar Transform

Listing 5: Sample Matlab code.

```matlab

function  b = 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
[i,k] = size(Y)
op=zeros(i,k);

Y = double(Y);
Lo_Lo = sqrt(0.25)*[1,1;1,1];
Hi_Lo = sqrt(0.25)*[1,-1,;1,-1];
Lo_Hi = sqrt(0.25)*[1,1;-1,-1];
Hi_Hi = sqrt(0.25)*[1,-1;-1,1];
pic_Lo_Lo = conv2(Y,Lo_Lo,'same')
pic_Lo_Hi = conv2(Y,Lo_Hi,'same')
pic_Hi_Hi = conv2(Y,Hi_Hi,'same')
pic_Hi_Lo = conv2(Y,Hi_Lo,'same')

pic_Lo_Lo = imresize(pic_Lo_Lo, 0.5);
pic_Lo_Hi = imresize(pic_Lo_Hi, 0.5);
pic_Hi_Hi = imresize(pic_Hi_Hi, 0.5);
pic_Hi_Lo = imresize(pic_Hi_Lo, 0.5);


op(1:i/2,1:k/2) = pic_Lo_Lo;
op(1:i/2,k/2+1:k) = pic_Lo_Hi    ;      %  row low freq , col Hi freq
op(i/2+1:i,1:k/2) = pic_Hi_Lo   ;       %  row Hi freq , col low freq
op(i/2+1:i,k/2+1:k) = pic_Hi_Hi    ;    %  all high
b = op;
```

The Haar transform is a very simple way to do the energy compression.It can be divided into 4 parts,LoLo,LoHi,HiLo and HiHi Subband. The LoLo subband includes the details of the original image. The LoHi Subband includes the horizontal details of the image. The HiLo Subband includes the Vertical details of the image and HiHi Subband includes the diagonal details of the image. Fig.3 can describe how it works.Totally, it can be described as these steps as followed.

- Step 1,Find the averaging. Calculate the average value of adjacent pixel pairs to get a new image with lower resolution, that is, the resolution of the new image is 1/2 of the original
- Step 2,Find the difference. Obviously, when this image is represented by 1/2 pixels of the original image, the information of the image has been partially lost. In order to be able to reconstruct the original image composed of full pixels from the image composed of 1/2 pixels, the detail coefficients of some images need to be stored in order to retrieve the lost information during reconstruction. The method is to subtract the average value of the pixel pair from the first pixel value of the pixel pair, or use the difference of this pixel pair to divide by 2.(HiLo and LoHi)
- Step 3, Repeat steps 1 and 2 to further decompose the image obtained in the first step into lower resolution images and detail coefficients.

The process of haar transform is equal to the 2-D convolution process. I set 4 different kernels to do the convolution calculation with the image to generate the subband image. The kernel is illustrated as below :

$$LoLo = \frac{1}{2} \times \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \tag{4}$$

$$HiLo = \frac{1}{2} \times \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix} \tag{5}$$

$$LoHi = \frac{1}{2} \times \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \tag{6}$$

$$HiHi = \frac{1}{2} \times \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{7}$$

- Q3B:Using the Matlab function that you have written to implement the 1-level Haar Transform, and using your assigned quantisation step size, calculate the resulting Entropy H_{qhaar} of the transformed image after quantisation has been applied

Calculate the overall entropy is trickier. Each coefficient in a band represents 4 pixels location in the initial image. So H_{qhaar} should be calculated as the equation shown below :

$$H(X) = \frac{LoLo}{4} + \frac{HiLo}{4} + \frac{LoHi}{4} + \frac{HiHi}{4} \tag{8}$$

Where the Denominator is the entropy of the subband image.

Listing 6: Sample Matlab code.

```
function entropy = calcEntropyofHaar(a,b,c,d)
A = calcEntropy(a);
B = calcEntropy(b);
C = calcEntropy(c);
```

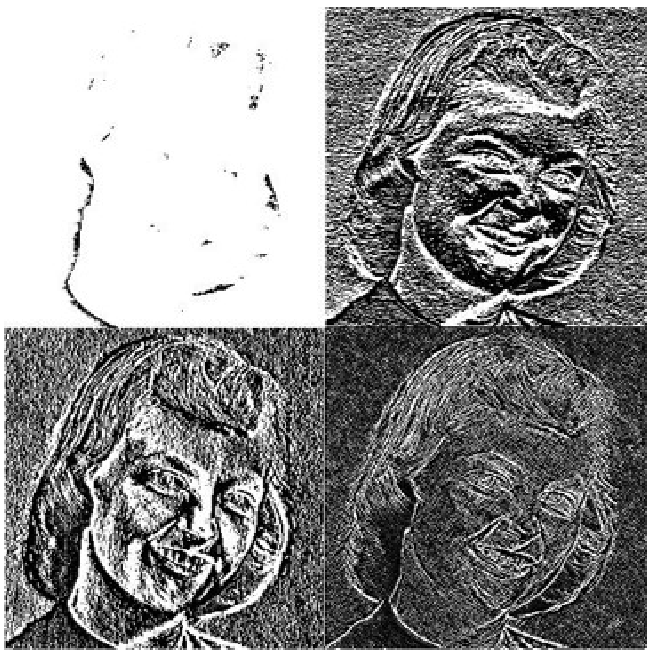

Figure 4: Haar Transform Level 1.

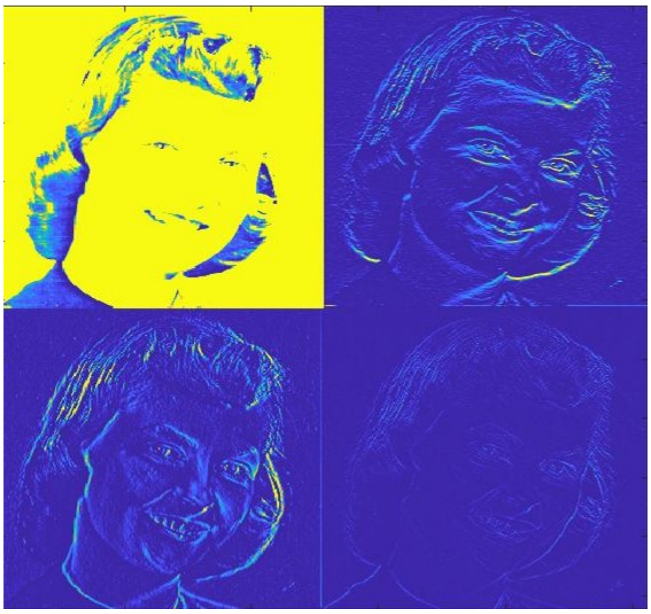

Figure 5: Haar Transform Level 1.

In order to illustrate the image more clearly,I change the color of the background.

```
D = calcEntropy(d);
entropy = (A+B+C+D)/4;
end
```

After using these function, i get the result of the H_{qhaar}. It shown as below.

2.4821 0.5063 0.3774 0.3201 0.3201 0.3201

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

The H_{qhaar} is much less than the H_{qi}, because the after the haar transform, it miss some ingredient of image. Here is the example for comparing of the image.

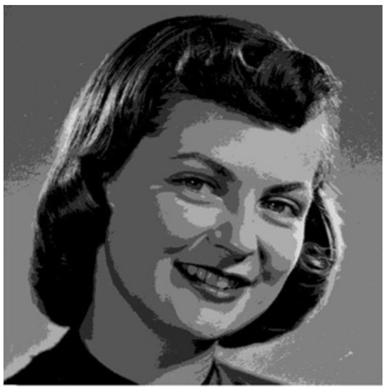

Figure 6: Original quantised,step = 43

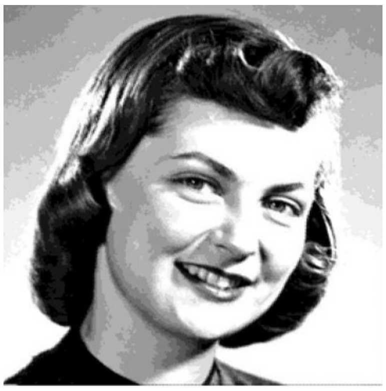

Figure 7: Haar quantised,step = 43

- Q3D To check the quality of the image after compression, the only thing to do is to reconstruct the picture using the calcInvHaar function. Compress and reconstruct your image using stepsizes of Qstep/2, Qstep and 2 × Qstep and comment on the differences you see in the pictures compared to the original image and the quantised image from Section 2.

After doing the Haar transform to the original image, I choose 44 as the quantisation step.using stepsizes of Qstep/2, Qstep and 2 × Qstep, the firgure above is the result.From the result of the entropy, we can see all the entropy is lower than the Quantised image before doing the Haar transform. With the quantisation step increasing, the entropy of the restructed image is

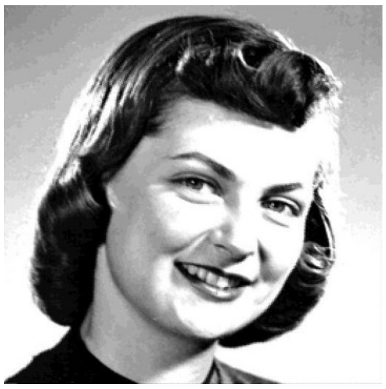

Figure 8: Haar restruct,step = 22

Entropy of Fig.8 is 0.3289

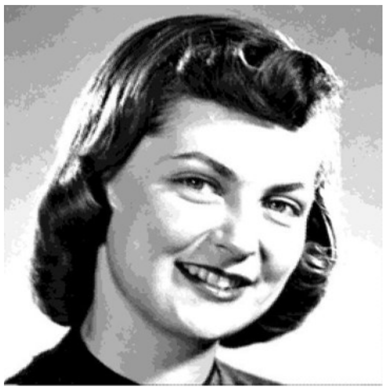

Figure 9: Haar restruct,step = 44

Entropy of Fig.9 is 0.4815

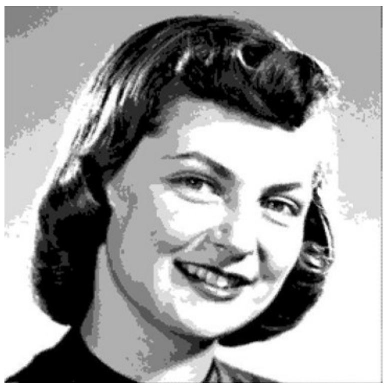

Figure 10: Haar restruct,step = 88

Entropy of Fig.10 is 0.6129

increasing. The lower quantisation restructed image shows better than the higher quantisation value image.The Fig.8 is more smoothly than others.

- Q3e Calculate the entropy of the transformed and quantised images and the mean squared error between each reconstructed image and the original image. Comment on the relationship between the entropy and the objective quality metric for the 3 quantisation step sizes.

By using the calcMSE function(where is shown on page3), i calculate the result of the MSE between the original image and the transform and quantised image. The result is shown as below:
The step 22 quantisation image with original of MSE:36535
The step 44 quantisation image with original of MSE:140199
The step 88 quantisation image with original of MSE:577940
With the step value increasing, the MSE value is increasing too, which means the smaller step of quantisation is more close to the original image.With the step increasing,the quality of image becomes worse and worse.

- 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). Does the MSE metric correctly rank the perceived quality of the 6 compressed images (3 quantisation levels with/without transformation into the Haar domain)? Explain your answer.

Without doing the Haar transform, I calculate the MSE value between the quantisation image and the original image. It is clearly shown on Fig.2. The quantisation step is 43. The MSE metric correctly ranks the perceived quality of the 6 compressed images.
Back to the transformation image, Fig.8 to Fig.10, the MSE value also ranks correctly the perceived quality of these image.

Problem 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.

Listing 7: Sample Matlab code.

```
function  x = calcHaar(pic,n)
% This function takes as input a 2D array Y containing
% the image intensities of a picture and returns the 1-level
% Haar Transform
% n is the number of levels used.
[i,k] = size(pic)
op=zeros(i,k);
```

```matlab

pic = double(pic);
% set up the kernel.
Lo_Lo = sqrt(0.25)*[1,1;1,1];
Hi_Lo = sqrt(0.25)*[1,-1,;1,-1];
Lo_Hi = sqrt(0.25)*[1,1;-1,-1];
Hi_Hi = sqrt(0.25)*[1,-1;-1,1];

pic_Lo_Lo = conv2(pic,Lo_Lo,'same')
pic_Lo_Hi = conv2(pic,Lo_Hi,'same')
pic_Hi_Hi = conv2(pic,Hi_Hi,'same')
pic_Hi_Lo = conv2(pic,Hi_Lo,'same')
% adjust the size
pic_Lo_Lo = imresize(pic_Lo_Lo, 0.5);
pic_Lo_Hi = imresize(pic_Lo_Hi, 0.5);
pic_Hi_Hi = imresize(pic_Hi_Hi, 0.5);
pic_Hi_Lo = imresize(pic_Hi_Lo, 0.5);

op(1:i/2,1:k/2) = pic_Lo_Lo;
op(1:i/2,k/2+1:k) = pic_Lo_Hi   ;     %  row low freq , col Hi freq
op(i/2+1:i,1:k/2) = pic_Hi_Lo  ;      %  row Hi freq , col low freq
op(i/2+1:i,k/2+1:k) = pic_Hi_Hi   ;   %  all high

 if ( n >1 )% different cases
  while n > 1;
    pic = pic_Lo_Lo;
    i = i/2 ;
    k = k/2 ;
pic_Lo_Lo = conv2(pic,Lo_Lo,'same')
pic_Lo_Hi = conv2(pic,Lo_Hi,'same')
pic_Hi_Hi = conv2(pic,Hi_Hi,'same')
pic_Hi_Lo = conv2(pic,Hi_Lo,'same')

pic_Lo_Lo = imresize(pic_Lo_Lo, 0.5);
pic_Lo_Hi = imresize(pic_Lo_Hi, 0.5);
pic_Hi_Hi = imresize(pic_Hi_Hi, 0.5);
pic_Hi_Lo = imresize(pic_Hi_Lo, 0.5);

op(1:i/2,1:k/2) = pic_Lo_Lo;
op(1:i/2,k/2+1:k) = pic_Lo_Hi   ;     %  row low freq , col Hi freq
op(i/2+1:i,1:k/2) = pic_Hi_Lo  ;      %  row Hi freq , col low freq
op(i/2+1:i,k/2+1:k) = pic_Hi_Hi   ;%  all high
n = n - 1;
  end
 else
   op = op
 end
```

```
imshow(op)
x =op;
%   Shida Sheng
```

This is the multi-level 2D Haar Transform. In order to know whether it works or not, I run the testHaar function to see the outcomes. It shows : calcHaar: OK and the outcomes image is shown below.

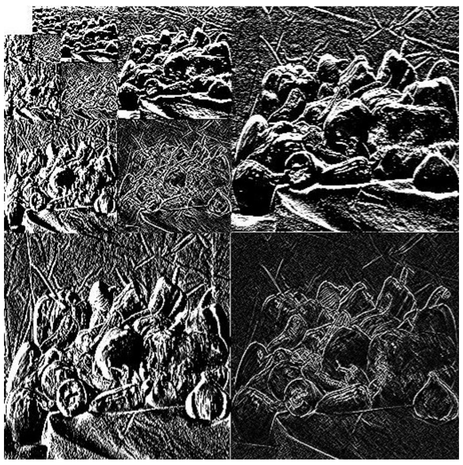

Figure 11: Test image

- 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.

I choose second level Haar transform to do this question.So first, by using the CalcHaar function, i generate the image below.

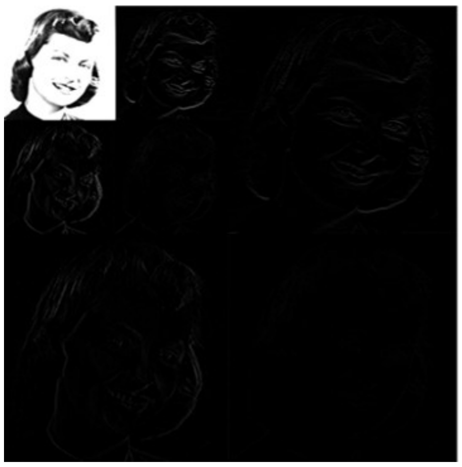

Figure 12: Second level Haar transform

I choose 30 as the Quantisation step, after doing the the quantisation, i will use the calinvHaar function to reconstruct the image.The reconstructed image is shown below:

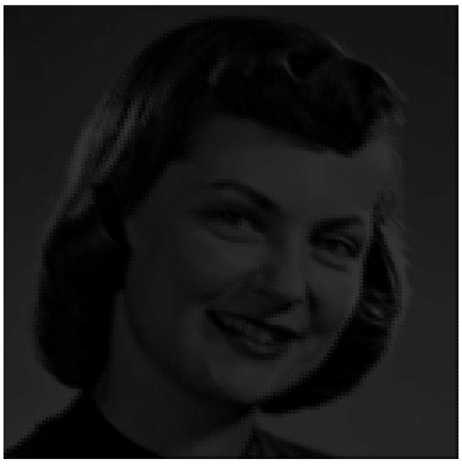

Figure 13: Second level Haar transform

Entropy: 4.1655 ,MSE between original image : 5515,Qstep = 30

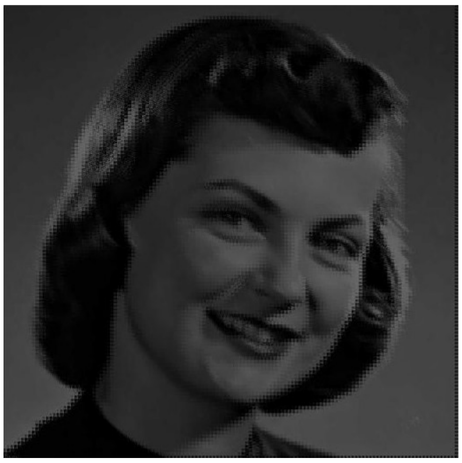

Figure 14: Second level Haar transform

Entropy: 5.1118,MSE between original image :2606,Qstep = 60

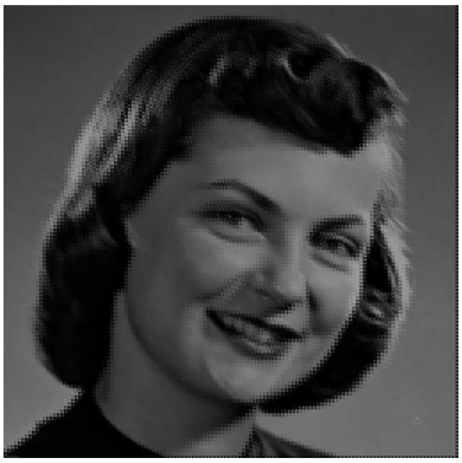

Figure 15: Second level Haar transform

Entropy: 5.6775,MSE between original image :876,Qstep = 90

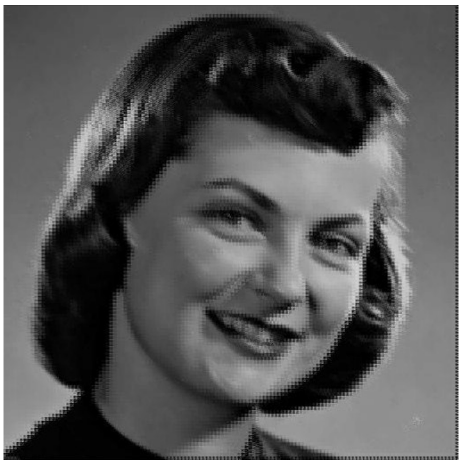

Figure 16: Second level Haar transform

Entropy: 6.0791,MSE between original image :348,Qstep = 120

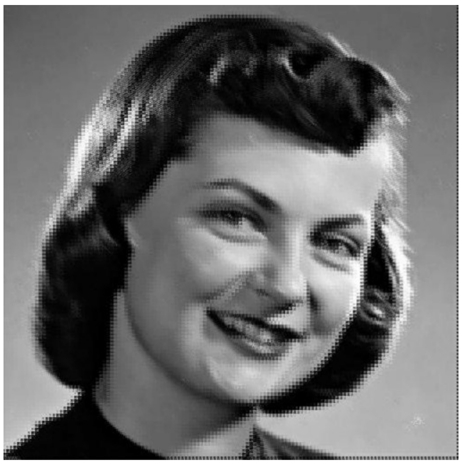

Figure 17: Second level Haar transform

Entropy: 6.3879,MSE between original image :1012,Qstep = 150

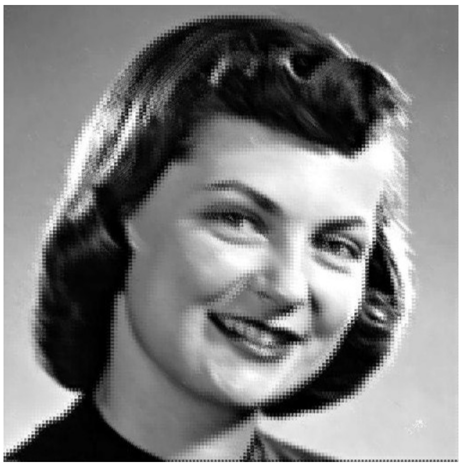

Figure 18: Second level Haar transform

Entropy: 6.5879,MSE between original image :2764,Qstep =180

By comparing the MSE value and the entropy value of the reconstructed image, We can know when the quantisation value becomes to 120, the quantised image can reconstruct very close to the original image,which means the quality of the quantised image is good. However, with the quantisation value, the quality of the image is going bad. That means if we want to reconstruct a good image, we have to find out the best quantisation value to do quantise to the image.

Summary

This lab let me have a straightly understanding of the image compression.By saving the details of the original image(high frequency details,low frequency details and so on.), we can compress a image easily. It also has some problems when i was doing this lab, i noticed that, after doing the Haar transform, the image value is big, so result will be very bright. After doing the transformation, we have to convert the result into Unsigned Integers of 8 bits value, it is very important.