Problem 3

I am not sure if I understand this problem correctly, I have tried my best.

(1) Pretreatment

First, I choose an image, and read it into MATLAB.

```
A=imread('C:\Users\Alex\Desktop\1.jpg');
imshow(A);
```

I filtered the b and g values out and keep the r value only. The image here is 1200*1920.

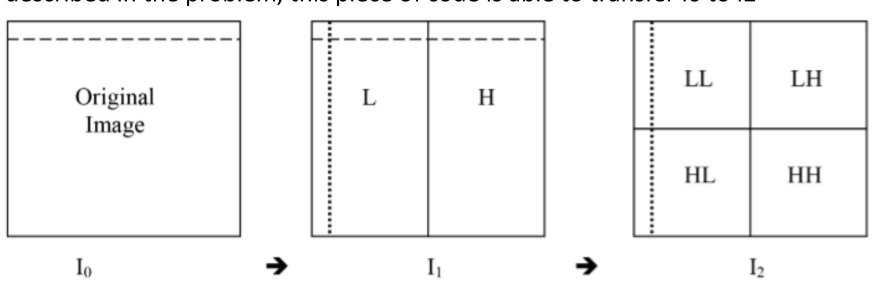
```
A=A(:,:,1);
imshow(A);
```

(2) Filter bank design and image processing

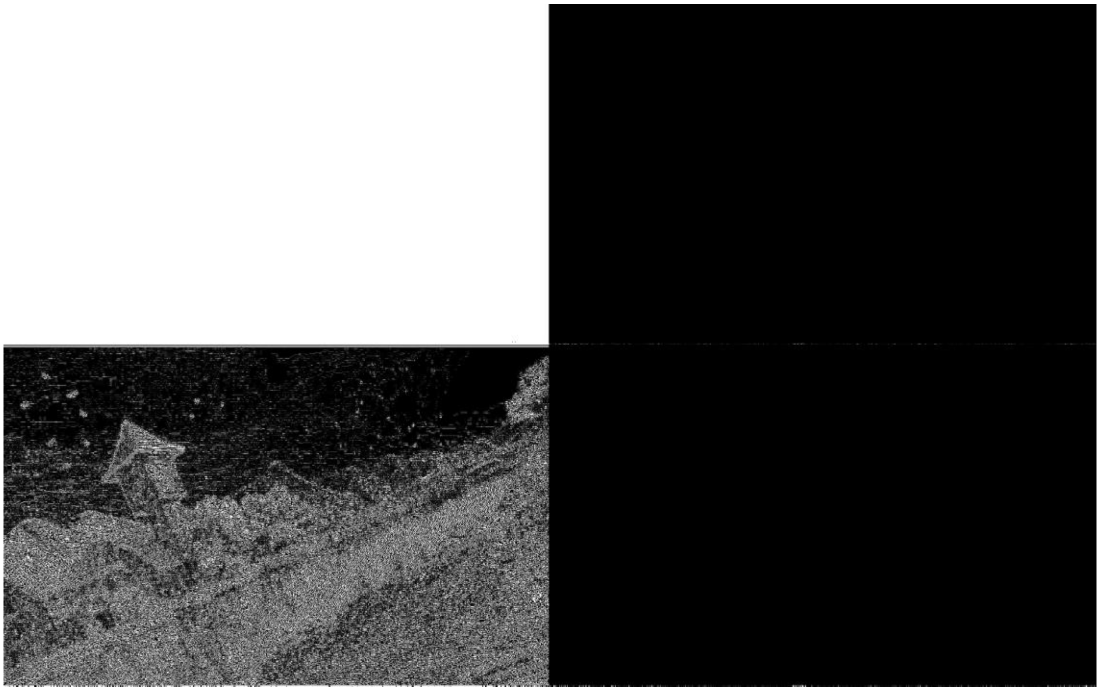
The MATLAB code of my filter bank is as follow. The input of this function is an image matrix. The output is the 4-section image matrix accordingly.

```
function [C] = QMF(A)
N    = 29;
Fc   = 0.5363563;
flag = 'scale';
Beta = 9;
win = kaiser(N+1, Beta);
h0  = fir1(N, Fc, 'low', win, flag);
n=rem(1:30,2);
m=1*(n>0.5)+(-1)*(n<=0.5);
h1=h0.*m;
for i=1:length(A(:,1))
    B_L=dyaddown(conv(A(i,:),h0),1);

B(i,1:ceil(0.5*length(A(1,:))))=B_L(ceil(0.25*length(h0)):(ceil(0.25*length(h0))+ceil(
0.5*length(A(1,:)))-1));
end
for j=1:length(A(:,1))
    B_H=dyaddown(conv(A(i,:),h1),1);

B(i,ceil(0.5*length(A(1,:)))+1:(2*ceil(0.5*length(A(1,:)))))=B_H(ceil(0.25*length(h0))
:(ceil(0.25*length(h0))+ceil(0.5*length(A(1,:)))-1));
end
for k=1:length(B(1,:))
    C_L_row=dyaddown(conv(B(:,k),h0),1);

C_L_row_adjust=C_L_row(ceil(0.25*length(h0)):(ceil(0.25*length(h0))+ceil(0.5*length(A(
:,1)))-1));
    C(1:ceil(0.5*length(A(:,1))),k)=C_L_row_adjust(:);
end
for l=1:length(B(1,:))
    C_H_row=dyaddown(conv(B(:,l),h1),1);

C_H_row_adjust=C_H_row(ceil(0.25*length(h0)):(ceil(0.25*length(h0))+ceil(0.5*length(A(
:,1)))-1));
    C(ceil(0.5*length(A(:,1)))+1:(2*ceil(0.5*length(A(:,1)))),l)=C_H_row_adjust(:);
end
imshow(C);
end
```

As described in the problem, this piece of code is able to transfer I0 to I2



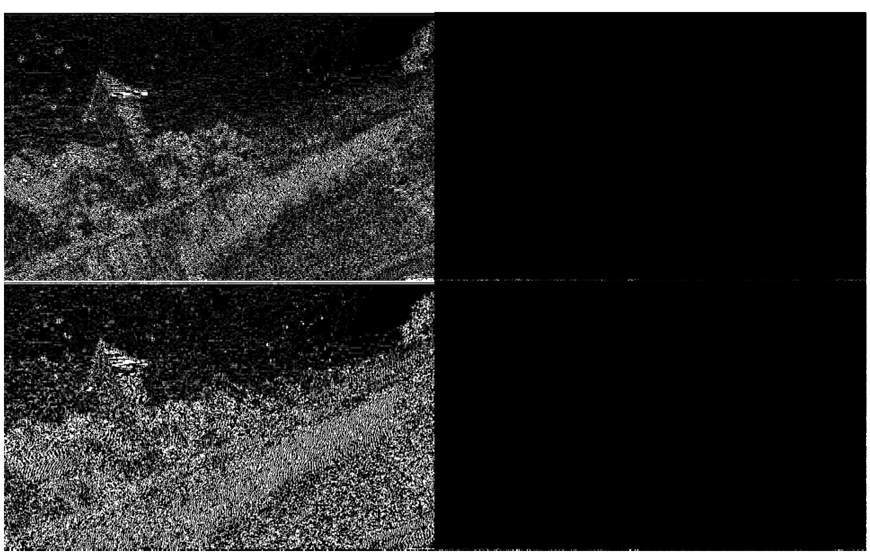
Then I started to deal with the image A that I have chosen above with this MATLAB function. And the result is shown below.



As we can see, only the HL section demonstrate a fuzzy image that looks like the original image A.

I decide to process the HL section a step further with the filter bank.

```
A_HL=A_1(601:1200,1:960);
A_HL_1=QMF(A_HL);
```



The LL and HL component of image A_HL_1 looks a little bit like the original picture.

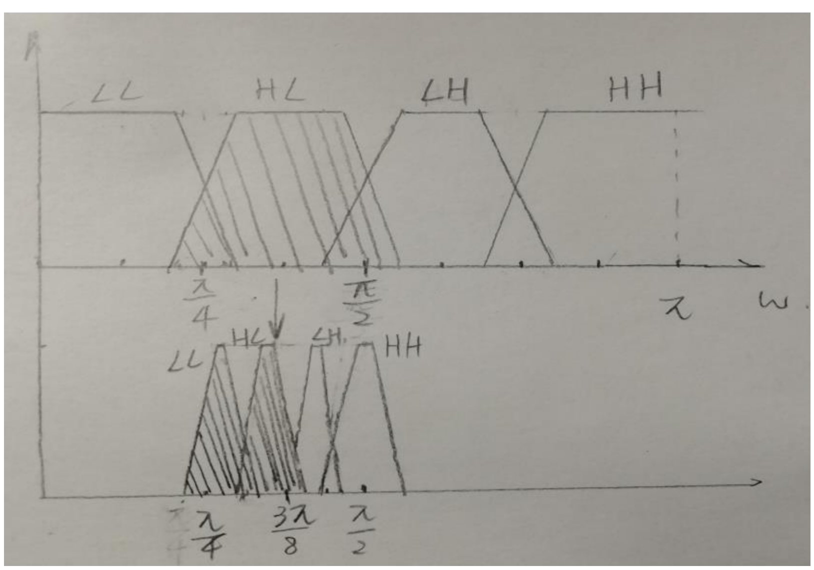
(3) Analysis and filter bank structure.

Analysis:

First, I want to confess that I cannot figure out why there is nothing but blank space in the LL region of the first result.

As far as I can understand, the high frequency signals are illustrated by the clean and distinct outlines in an image, while the low frequency is shown in those monotonous and smooth areas. As for my image, it is reasonable that the LH and HH regions are black.

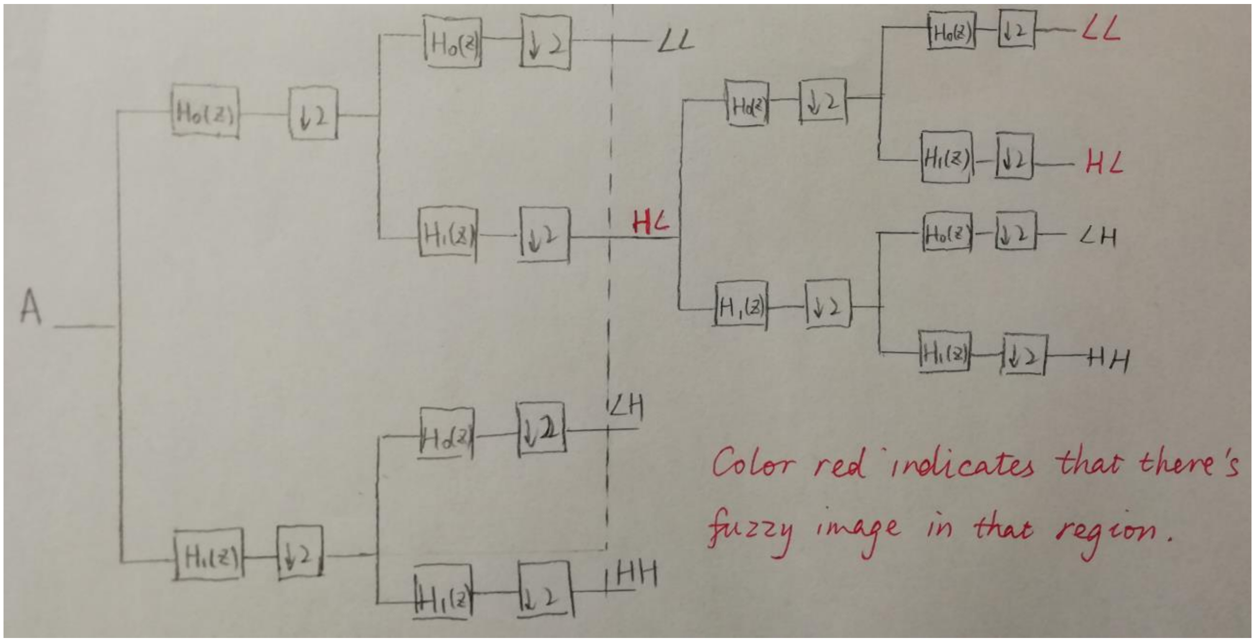
Let's just assume there is no LL component in this image(unlikely). The fuzzy images come from the two results that look like the original give me a rough understanding of the frequency distribution of this image.



The frequency of the chosen image is mainly concentrated in the shadowed area: $(\frac{\pi}{4}, \frac{3\pi}{8})$.

Filer bank structure

The filter bank tree structure is as follow.



I started quantization from this point.

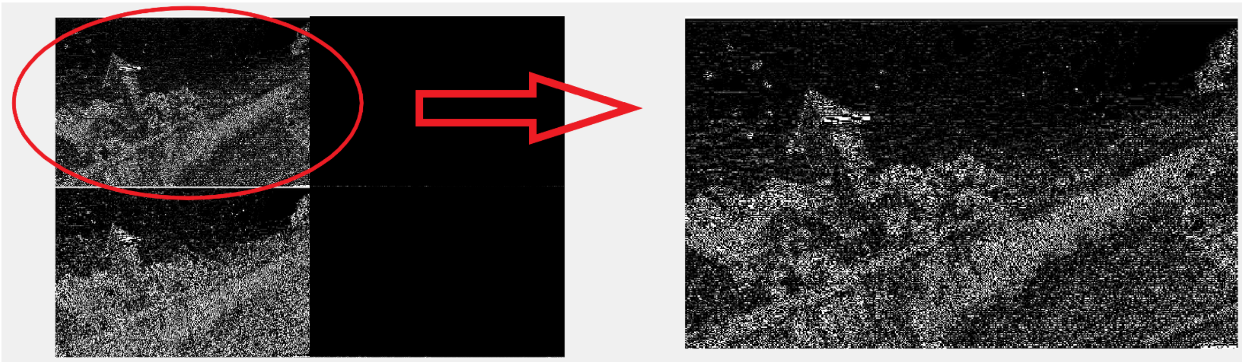
(4) Quantization.

My quantizer MATLAB function is as follow. The input are image vector and quantization step.

```
function [x_hat]=qtest(x,q_step)
x_c=round(x/q_step);
x_hat=x_c*q_step;
imshow(x_hat);
end
```

Again, I am not sure if my understanding of quantization is accurate. I assume that the reason why the average bits per pixel is 8 is that the values of image matrix elements are ranged from 0 to 255. 2^8 equals to 256. As a result, if we choose binary representation, it takes 8 bits.

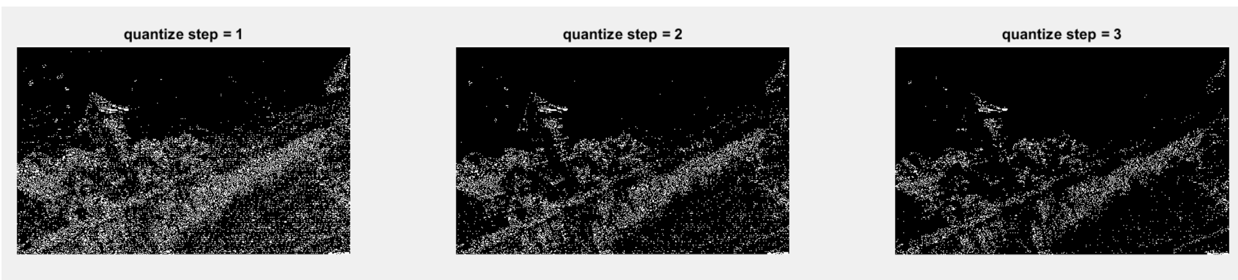
So, what I did is to separate LL section of my second filter bank result (region in the red circle), and find its element value range.



```
A_HL_LL=A_HL_1(1:300,1:480);  % Separate the region

max(max(A_HL_LL)) % Find the largest element.
```

Value of the largest element is 38.6629. This means I need at most 6 bits to represent this subband image.

Then I use my quantizer above on this subband image to find a proper quantize step. Here are the results for quantize step of 1, 2 and 3.



As the step goes from 1 to 3, the image is getting less messy. I believe the reason for this is that the quantizer "filters out" some high-frequency component in this process.

If I choose 2 as my quantize step, then I need 5 bits to represent this subband image in binary.

This is as far as I can go in this problem.