Lab 03

Haar Transform

Aaron Dinesh - 20332661

02 March 2024

Declaration

Signed: Aaron Dinesh Date: 02 March 2024

1 Image Entropy

1.1 Writing an Entropy Function

Given a random variable $\mathcal{X}$ that follows a particular distribution, the entropy of that variable, denoted $H(\mathcal{X})$, is given by:

$$H(\mathcal{X}) = -\sum_{x \in \mathcal{X}} p(x) \log_2(p(x)) \tag{1}$$

In image processing, it is given the unit bits/pixel. Special care must be applied when the probability of the symbol is 0. By definition, we say that this has 0 entropy. In MATLAB, this was defined as follows:

```
function entropy = calcEntropy(Y)
    entropy_temp = 0;
    [hist, ~] = histcounts(Y, 2048, 'Normalization','probability');
    for i = 1:1:length(hist)
        if hist(i) == 0
            continue
        end
        entropy_temp = entropy_temp - hist(i)*log2(hist(i));
    end
    entropy = entropy_temp;
end
```

1.2 Quantizing the Image

Quantization can be performed using the following formula, where Q_{step} defines the quantization step.

$$I_{quant} = round(\frac{I}{Q_{step}}) \tag{2}$$

The reason for Quantization is that we end up with fewer symbols which leads to a lower entropy.

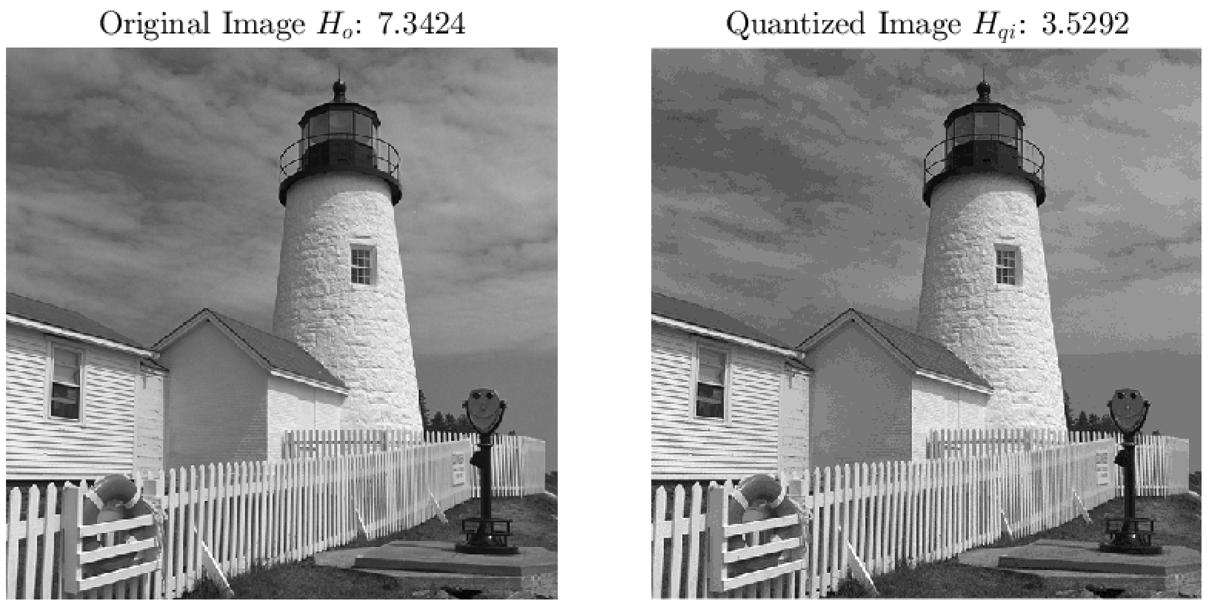
Original Image H_o: 7.3424
Quantized Image H_{qi}: 3.5292

Figure 1: Original and Quantized version of Kodim19

1.3 PSNR and Visual Quality

The Peak Signal-to-Noise Ratio is a common metric used by many people to indicate the quality of a photo after some type of transformation. It works by comparing the transformed photo to a reference image, with low values indicating large differences from the reference and high values indicating that the two images are very similar. This can be calculated in MATLAB by calling the psnr function and

passing the transformed image and the reference image as parameters. This yielded a result of 5.845 for the quantized image.
Visually the quantized image looks worse than the reference image. Due to the quantization, we lose the fine transitions between similar colors in the sky and on the walls. Instead, similar colors are replaced with one color, leading to the patches of homogeneous color we see in the image.

2 2D Haar Transform

2.1 Implimentation

The level 1 Haar Transform was implemented in the following way

```matlab
function H = calcHaarLevel1(Y)
    if (mod(size(Y,1),2) ~= 0)
        error('height must be multiple of 2');
    end
    if (mod(size(Y,2),2) ~= 0)
        error('width must be multiple of 2');
    end

    % Calculate the 4 parts the intermidate matrix
    a = Y(1:2:end, 1:2:end);
    b = Y(1:2:end, 2:2:end);
    c = Y(2:2:end, 1:2:end);
    d = Y(2:2:end, 2:2:end);

    % Mix them to create the 4 'entries'
    lolo = a + b + c + d;
    hilo = a - b + c - d;
    lohi = a - c + b - d;
    hihi = a - b - c + d;

    % Concat them togetht to form the transformed image
    H = (1/2)*vertcat(horzcat(lolo, hilo), horzcat(lohi, hihi));
end
```

2.2 Quantizing the Haar Transform

Using the function implemented above, I calculated the level 1 Haar Transform of the Kodim19 image. The output of the transform was then quantized using the same procedure as before with a Q_{step} of 15. The inverse transforms were then calculated and are shown below in Figure 2

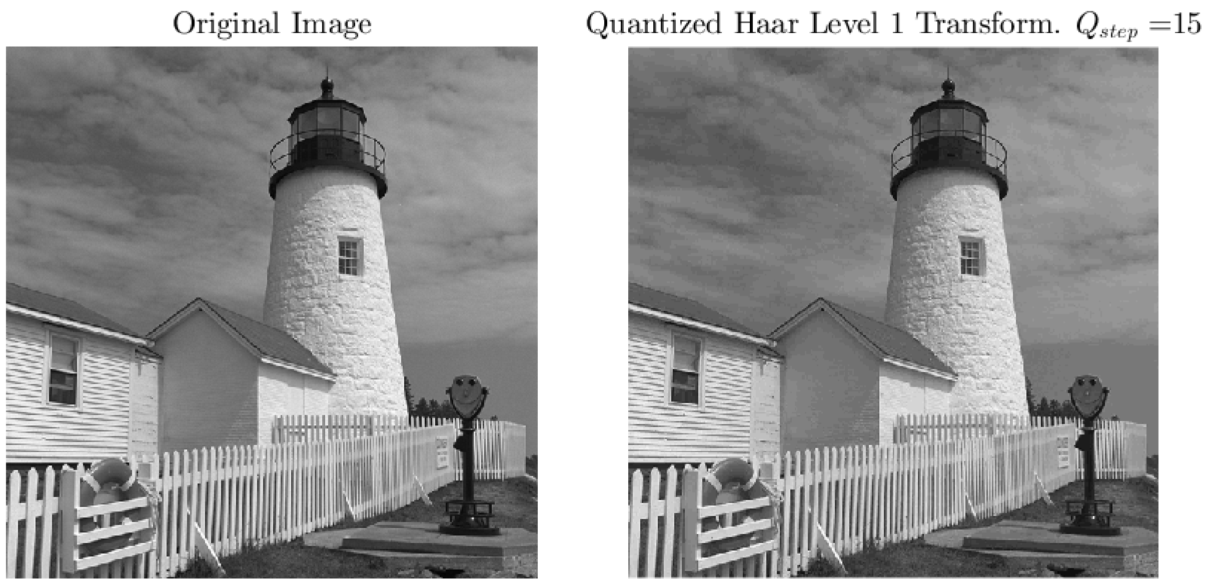

Figure 2: Haar and Quantized Haar Transform of Kodim19

The entropy calculated for the quantized haar transform of kodim19 was 1.96687 bits/pixel.

2.3 Comparing H_{qharr} and H_{qi}

H_{qharr} denotes the entropy of the quantized level 1 haar transform while H_{qi} denotes the entropy of the quantized image. We can clearly see that the entropy of the quantized Haar transform is a lot lower than just performing normal quantization, 1.96687 as opposed to 5.845 bits/pixel. This is because of the energy compaction property of the Haar transform. By applying the transform we essentially concentrate the probability distribution around a limited number of symbols. This has the effect of lowering the overall entropy. Quantizing the symbols then further concentrates the energy leading to the lower entropy.

2.4 Visually Comparing The Quantized Haar Transform and The Original Images

Visually, the quantization artifacts are a lot harder to see in the quantized Haar image in Figure 2 when compared with the quantized image in Figure 1. Looking closely, we can still see some quantization artifacts in the form of small patches of homogeneous colors. But the effect is a lot more subtle here than in Figure 1.

3 Rate-Distorsion Curve

3.1 RD Curve for No Transformation

Another important metric that many people use in image processing is the Rate-Distortion curve. This relates the entropy of an image at a particular Q_{step} to the PSNR. Here I considered $Q_{step} \in [2, 4, 8, 16, 32, 64, 128]$ with no transformation the results of which are shown in Figure 3.

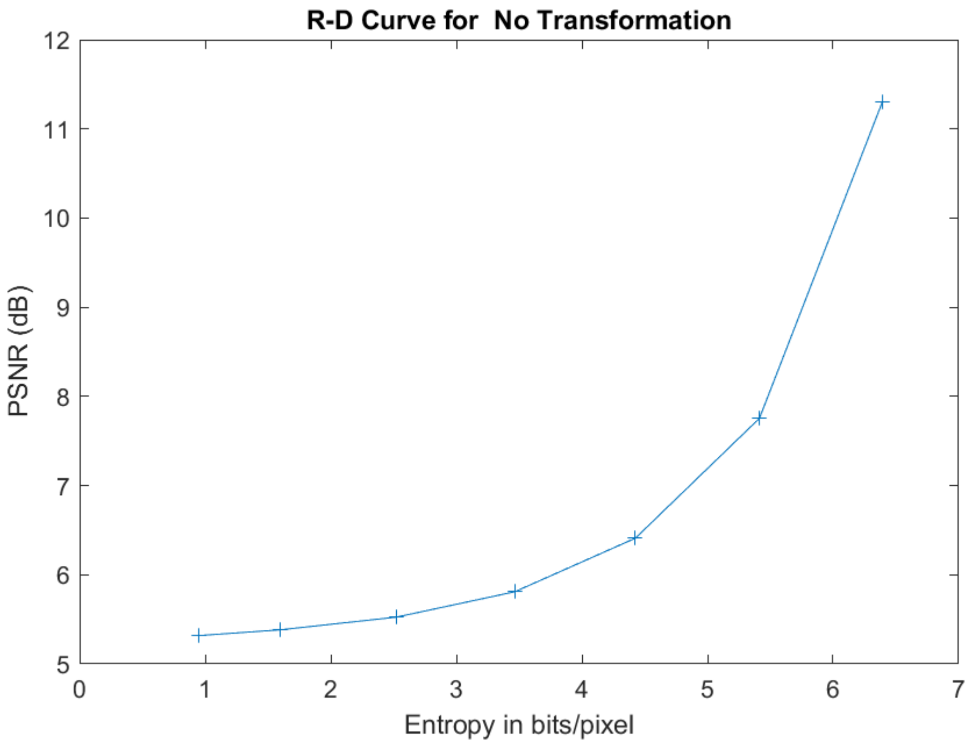

Figure 3: Rate-Distorsion Curve for No Transform

3.2 RD Curve with Level 1 Haar

Here I added data points for the level 1 Haar transform.

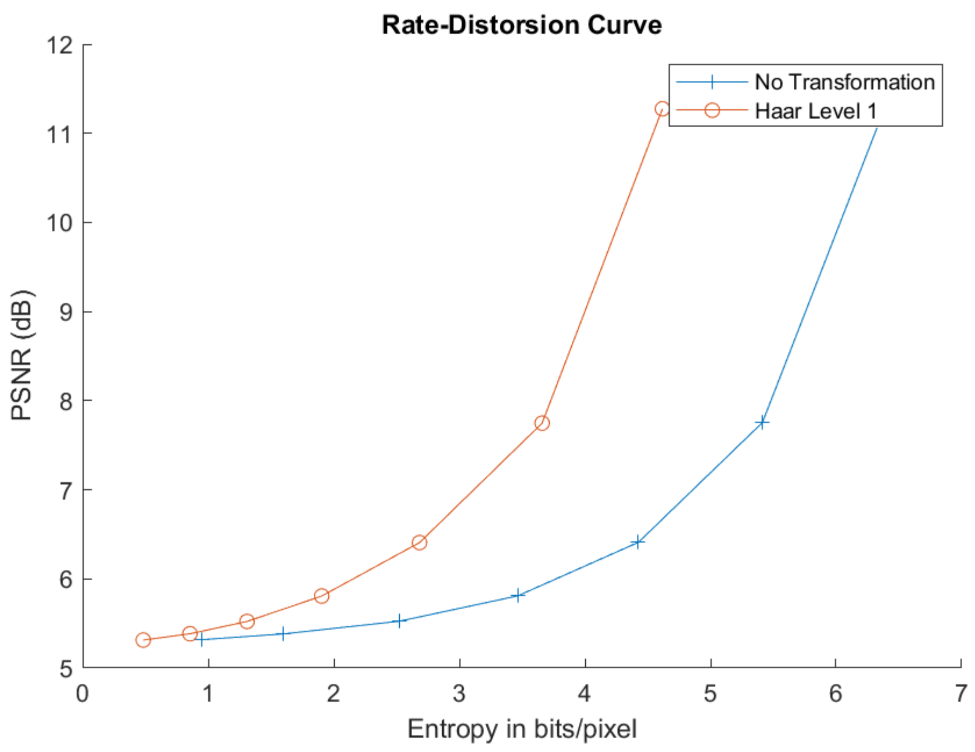

Figure 4: Rate-Distorsion Curve with Level 1 Haar

3.3 Comparing Transform with No Transform

Judging, by the PSNR value (high is better) we can clearly see that the Haar transform even at level 1 offers us lower bitrates at higher quality when compared with no transform at the same quantization level. This is ideal for sharing images over the internet where we want the highest quality for the lowest bitrate.

4 Multi-Level 2D Haar Transform

4.1 Implimentation

After performing the Haar transform, one can notice that the ”LoLo” section of the transform is largely similar to the original image. As such some gains can be achieved by applying the Haar transform recursively to the LoLo section of the transform. The n-point Haar Transform is then defined by recursively applying the Haar Transform to the LoLo section of the level 1 transform $n-1$ times. The implementation is shown below.

```
function H = calcHaar(I, n)
    if (mod(size(I,1),2) ~= 0)
        error('height must be multiple of 2');
    end
    if (mod(size(I,2),2) ~= 0)
        error('width must be multiple of 2');
    end

    % Calculate the level 1 Haar transform
    Itemp = calcHaarLevel1(I);
    xSize = size(Itemp, 2);
    ySize = size(Itemp, 1);

    % Apply the Haar Transform to the LoLo section of the resulting matrix.
    % This corresponds to applying the Haar transform to the upper left
    % quarant of the matrix.
    for i = 1:n-1
        Itemp(1:ySize/(2^i), 1:xSize/(2^i)) = calcHaarLevel1(Itemp(1:ySize/(2^i), 1:
    xSize/(2^i)));
    end

    H = Itemp;
end
```

4.2 RD curves for Level 1-5 Haar Transform

Shown below in Figure 5 we can see the RD curves for the Q_{step} values mentioned before.

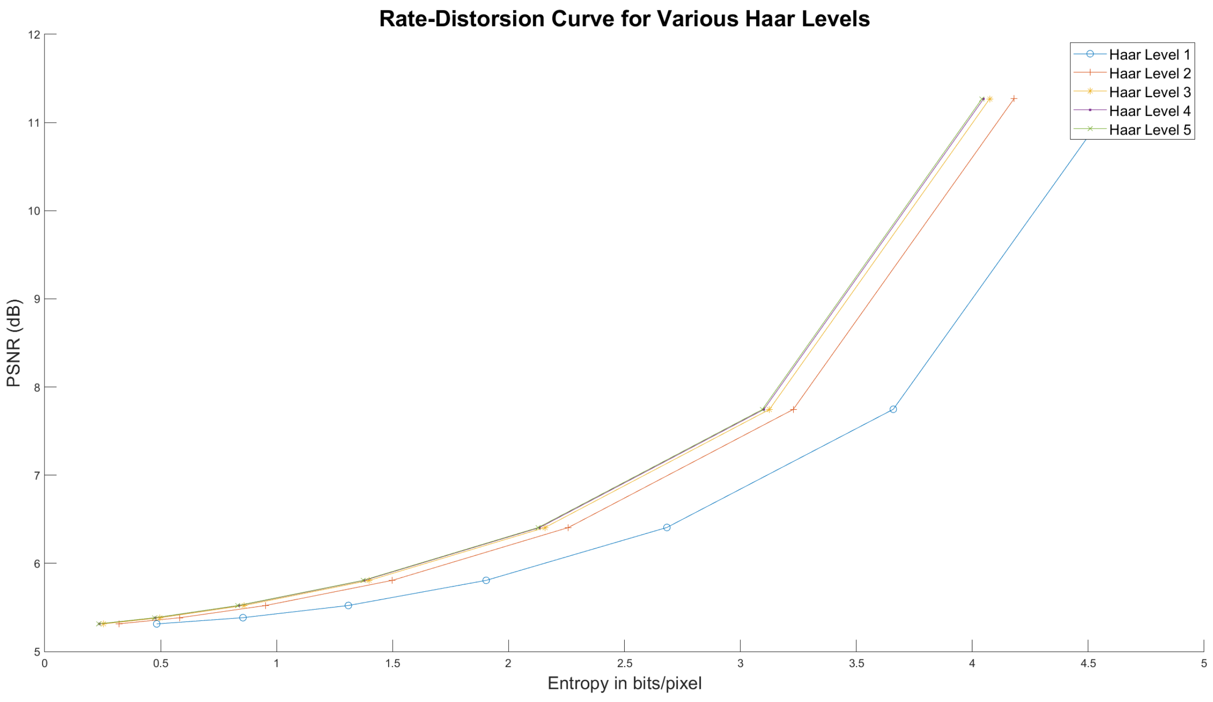

Figure 5: RD Curve for Multiple Haar Levels

Looking at the curves we see clear diminishing returns as the Haar level increases. Past Haar level 3, the reduction in entropy isn't worth the extra processing time required to calculate it.