CMPE 362 INTRODUCTION TO SIGNAL PROCESSING FOR COMPUTER ENGINEERS

HW #4 Watermarking System Design

Deadline 16th May 2016 23.59

Motivation and Introduction

Watermarking is a technique through which the secure information is carried without degrading the quality of the original signal. In other words you will embed data to a multimedia file without affecting host multimedia file much. Watermarking is a frequently used signal processing application.

In this homework, you will design a watermarking system. You will implement your watermarking system on time domain, wavelet domain and frequency domain respectively. Wavelet domain is a hybrid domain between time and frequency domains. You will use the simplest Wavelet Transform Algorithm which is Haar Wavelet as described in the homework. You will simply follow the instructions you are given and calculate SNR and MOS (Mean Opinion Score) values for each watermarking scheme. The watermarked image is MxN pixels. The audio that watermark is embedded is a wav file. In order to embed the image to the audio file, the image and audio should be preprocessed.

Your image file is a 64x64 wm.bmp. Your host audio file is son.wav. You can find these files in HW4_Materials.zip. In other words, you will embed the image file into audio file.

1) Watermark on Time Domain *30 pts*

In this part, you will embed the data on time domain. Steps are as follows:

1.1) Preprocessing

1. Represent the grey-scale image watermark as a two dimensional *MxN* matrix. Each value of this matrix represent the pixel value which is between 0-255. Imread() method at MATLAB is used.

2. Convert this matrix into a one dimensional array. The rows will be concatenated after each other. This created array has MxN length.

3. Zeros and ones are needed in order to apply spread spectrum to embed different parts of audio bytes. First convert the pixel value into 8 bit binary number. Then, a new array to hold this zeros and ones is created. This new watermark array has a length that 8 times the old one. This new array’s elements will be embedded to audio.

4. Frequency hopping spread spectrum technique is used. Hopping sequence is necessary for this purpose. Length five hopping sequence is created where every element of this sequence

is an integer between 1-8. This sequence will tell us which bit to embed the watermark. ‘n’ means embed the watermark to the nth least significant bit.

5.Convert an audio signal into byte array.

1.2) Watermark Embedding

1.Determine the position that the watermark bit will be embedded by using hopping sequence. This hopping sequence will be repeated. For example if sixth bit will be embedded and the length of the hopping sequence is five, this bit will be embedded to the position that first element of hopping sequence specifies.(6mod5)

2.Obtain the embedded watermark bit from binary array.

3. Embed every one byte in four bytes of the audio file.

4. If the element is ‘0’ use bitwise ‘AND’. Otherwise use bitwise ‘OR’ with the audio byte arrays specified bit. This means replacing the specified bit in the audio byte with the watermark bit. For instance if the embedded bit is ‘1’, embedding will be done as follows:

audioBytes(i)=bitor(audioBytes(i,2^{n-1})

If the bit is zero:

audioBytes(i)=bitand(audioBytes(i,255-2^{n-1})

where audioBytes is the audio array, n is the bit’s embedding position.

5. Obey the hopping sequence periodically until the watermark bit array finishes.

1.3) Watermark Extraction

1. In the watermarked audio byte array, embedded bits location is obtained by using hopping sequence.
2. Retrieve the embedded bit by applying bitwise AND operation with the binary number which has all zeros but a one in the embedding position.
embeddedBit=bitand(audioBytes(i), 2^{n-1})/ 2^{n-1}

3. Put the retrieved bits into bit array.
4. Convert these binary representations into decimal representations. This operation will result in the pixel value array (M x N).
5. Convert pixel value array as an M x N two - dimensional matrix.

6. Represent this resulting 2D matrix as a gray scale image. Mat2gray() method in MATLAB is used.

2) Watermark on Wavelet Domain *30 pts*

Wavelet domain is another domain on signal processing. It can hold both time and frequency domain information. You will use the simplest wavelet algorithm Haars Wavelet. Steps for this scheme are as follows:

2.1) Preprocessing

1. The grey-scale image watermark is represented as a two dimensional *MxN* matrix. Each value of this matrix represent the pixel value which is between 0-255. Imread() method at MATLAB is used.

2. Convert this matrix into a one dimensional array. The rows will be concatenated after each other. This created array has MxN length.

3. Zeros and ones are needed in order to apply spread spectrum to embed different parts of audio bytes. First convert the byte value into 8 bit binary number. Then, a new array to hold this zeros and ones is created. This new watermark array has a length that 8 times the old one.

4. Length five hopping sequence is created where every element of this sequence is an integer between 1-8. This sequence will tell us which bit to embed the watermark. 'n' means embed the watermark to the nth least significant bit.

5.Convert an audio signal into byte array.

6. 2 level DWT is applied to audio byte array.1D Haar Wavelet Transform is used for this purpose.[9] High pass and low pass values are created like the following:

$$H1(i) = \frac{S(2i) - S(2i+1)}{2}$$

$$L1(i) = \frac{S(2i) + S(2i+1)}{2}$$

where S is the audio signal. The second level transform is:

$$H2(i) = \frac{L1(2i) - L1(2i+1)}{2}$$

$$L2(i) = \frac{L1(2i) + L1(2i+1)}{2}$$

7.Audio byte array is arranged so that L2- H2- H1 is concatenated after each other respectively.

2.1) Watermark Embedding

1.Determine the position that the watermark bit will be embedded by using hopping sequence. This hopping sequence will be repeated. For example if sixth bit will be embedded and the length of the hopping sequence is five, this bit will be embedded to the position that first element of hopping sequence specifies.(6mod5)

2.Obtain the embedded watermark bit from binary array.

3. Embed every one byte in four bytes of the audio file.

4. If the element is ‘0’ use bitwise ‘AND’. Otherwise use bitwise ‘OR’ with the audio byte arrays specified bit. This means replacing the specified bit in the audio byte with the watermark bit. For instance if the embedded bit is ‘1’, embedding will be done as follows:

`audioBytes(i)=bitor(audioBytes(i,`2^{n-1}`)`

If the bit is zero:

`audioBytes(i)=bitand(audioBytes(i,255-`2^{n-1}`)`

where audioBytes is the audio array, n is the bit’s embedding position. [8]

5. Obey the hopping sequence periodically until the water- mark bit array finishes. [8]

6. Apply 2-level Inverse DWT to the audio file.

$$L1(2i) = L2(i) + H2(i)$$

$$L1(2i+1) = L2(i) - H2(i)$$

$$S'(2i) = L1(i) + H1(i)$$

$$S'(2i+1) = L1(i) - H1(i)$$

S' is the watermarked original signal.

2.3) Watermark Extraction

1. Apply 2 level DWT to the S'.

2. In the watermarked audio byte array, embedded bits location is obtained by using hopping sequence. [8]

3. Retrieve the embedded bit by applying bitwise AND operation with the binary number which has all zeros but a one in the embedding position. [8]

`embeddedBit=bitand(audioBytes(i),` 2^{n-1}`)/`2^{n-1}

4. Put the retrieved bits into bit array. [8]

5. Convert these binary representations into decimal representations. This operation will result in the pixel value array (M x N). [8]

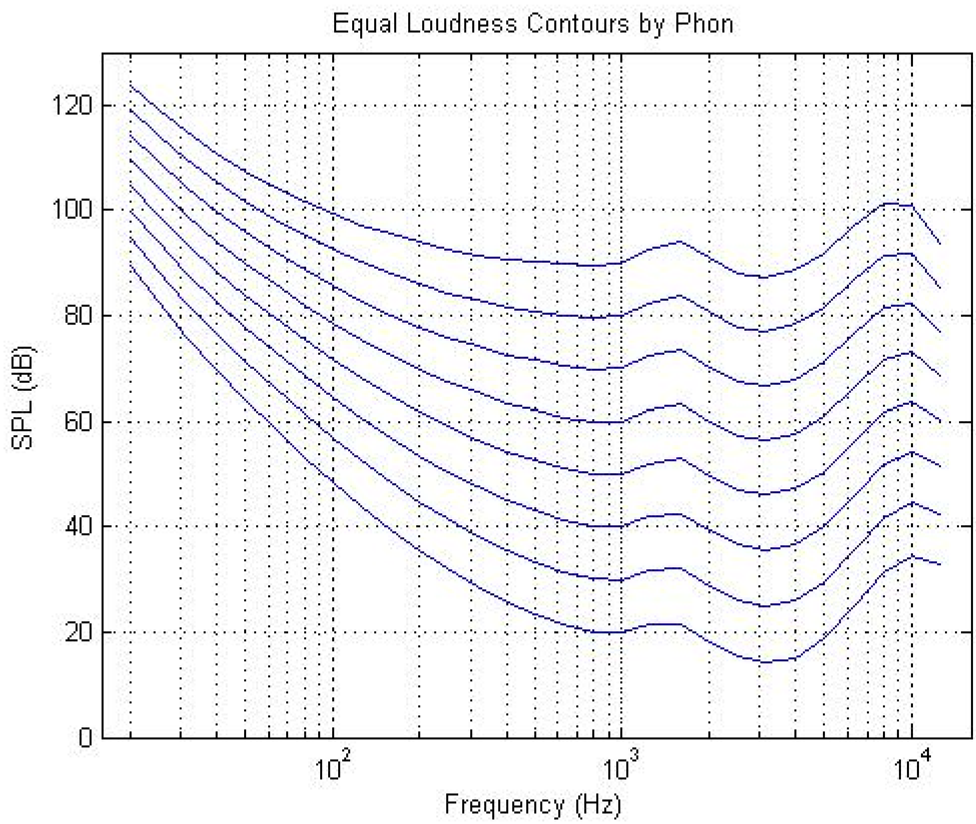
6. Convert pixel value array as an M x N two -dimensional matrix. [8]

7. Represent this resulting 2D matrix as a gray scale image. Mat2gray() method in MATLAB is used for this purpose. [8]

3) Watermark on Frequency Domain *40 pts*

In this part, you will convert the audio to frequency domain first. Look for the 0-20kHz band. (Hint: Use fft and create 20k bins. Then calculate amplitude for each bin.)

Below you see the equal loudness curve in Psychology of Human Sound Perception.



This curve explains the human sound perception. For example at 1Hz it is higher than 1 kHz. This means that you have to speak louder if you speak at 1Hz than 1 kHz so that you are heard equally.

For watermarking scheme if you embed the watermark to the high points, it would not be perceived much. In above figure 0-100Hz and 7200-10kHz are the highest points. The simplest algorithm would be:

1. The grey-scale image watermark is represented as a two dimensional *MxN* matrix. Each value of this matrix represent the pixel value which is between 0-255. Imread() method at MATLAB is used.

2. Convert this matrix into a one dimensional array. The rows will be concatenated after each other. This created array has MxN length.

3. Your image is 64 x 64. You have 2716 pixels. You can simply add the pixel values to abovementioned frequency band amplitude values (you can calculate the amplitudes by applying abs function to fft output) which have relatively higher decibel values.

For this question, you can come up with embedding algorithms that will lower SNR and increase MOS values with respect to simplest algorithm. In that case, you will get 20 points bonus.

4) Expected Outputs and Clarifications for Each Part (Time, Wavelet and Frequency Domains)

- You will implement the algorithms above and calculate the SNR value for each algorithm.
- Note that SNR will be calculated between the watermarked audio and original audio. (The audio before extraction step)
- You will not extract watermark for third case. (Freq. domain)
- You will listen the watermarked audio and original audio. You will give same value to your algorithms output. MOS values would be used.

MOS	Quality	Impairment
5	Excellent	Imperceptible
4	Good	Perceptible but not annoying
3	Fair	Slightly annoying
2	Poor	Annoying
1	Bad	Very annoying

- MOS values are subjective. Your grade will not be affected by low MOS values. However, you have to be as consistent as possible between your algorithms. Try to measure how much the audio signal is degraded.
- You will give MOS value to every watermarking scheme. Again you will use the audio before extraction when comparing with the original.
- (FOR BONUS) If you implement another algorithm than specified one in part 3, explain your algorithm and reasoning. Implement the specified algorithm and yours. Compare both SNR values and MOS values for both algorithms.

5) Report and Submission

Prepare a report explains your code briefly. Add the SNR results and MOS values for each three schemes to the report. Compare SNR values and MOS values for each scheme and make comments about relationships. Do the results come up as expected? Interpret the results for these questions. Compress the report (pdf) and the code files. Name it as "YourNumber_CmpE362_HW4.zip"(or rar, or 7z etc.). Send the file to yektasaid.can@gmail.com before the deadline. Subject of the mail would be CmpE362 HW4.

- If you implement the bonus part, do not forget to add comparison between specified algorithm and your algorithm to your report. (Part 3)