Homework 4 - Student Report
CMPE 362 Spring 2016

Batuhan Demir - 2013400159

May 16, 2016

1 Introduction

The topic of the of homework was quite interesting and has usage in real world. So it was important for me to do this project as good as i can. It helped to have an idea about data hiding techniques in general. But in bit setting and getting parts were challenging for me.

2 Questions & Answers

2.1 Watermark on Time Domain

In the code i tried to implement given procedure. Though i did not implement same algo, my implementation was quite similar and i think i get nearly expected working script. Result of the program which is 'lsb_watermarked.wav' audio file unfortunately is "wizzy". I think it is because frequency hopping.

- I give 2 points for the MOS value.
- My SNR value was 1.454.

I think my SNR value is consistent with wizzy audio file. It should be higher to hear more clear sound. And i think I gave consistent MOS value to watermarked audio, it is also low.

```
% % % % % % % Watermark on Time Domain % % % % %% %

% % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % PREPROCESSING % % %  %% % % % % %
% % % % % % % % % % % % % % % % % % % % % % % % %
file_name='son.wav';
[cover_audio,Fs]=audioread(file_name);
file_name='wm.bmp';
message=imread(file_name);

% Represent the grey-scale image watermark as a two dimensional MxN matrix
message=double(message);
message=round(message./256);
message=uint8(message);

% Read dimensions of the cover audio
Mc=size(cover_audio,1);

% Read dimensions of the message picture
Mm=size(message,1);
Nm=size(message,2);
message1=reshape(message,1,[]);

% % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % Watermark Embedding % % %  %% % % % % %
% % % % % % % % % % % % % % % % % % % % % % % % %

% This new a r r a y s elements will be embedded to audio

% Embed every one byte in four bytes of the audio file.
for ii = 1:4:Mc
        watermark1(ii)=message1(mod(ii,4)+1);
end
watermarked_audio=cover_audio;

% If the element is    0    use bitwise   A N D  .  Otherwise use bitwise   O R
% with the audio byte arrays specified bit.
```

```
for ii = 1:4:Mc
        watermarked_audio(ii)=bitset(cast(watermarked_audio(ii),'uint8'),1,watermark1(ii));
end
audiowrite('lsb_watermarked.wav',watermarked_audio,Fs);

% % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % Watermark Extraction % % %  %% % % % % %
% % % % % % % % % % % % % % % % % % % % % % % % % % %

file_name='lsb_watermarked.wav';
[watermarked_audio,Fs]=audioread(file_name);

% get size of watermarked audio
Mw=size(watermarked_audio,1);

% Retrieve the embedded bit by applying bit operation
for ii = 1:4:Mw
        watermark(ii)=bitget(cast(watermarked_audio(ii),'uint8'),1);
end

% Convert pixel value array as an M x N two - dimensional matrix
watermarkex= reshape(watermark,64,64);
% Represent this resulting 2D matrix as a gray scale image.
% Since i dont have Image Processing Toolbox i cannot use Mat2gray function

%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%% SNR Value%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%

s = mean(cover_audio.^2)/mean((cover_audio-watermarked_audio).^2);
snr = 10.^log(s);
```

2.2 Watermark on Wavelet Domain

In this question i was thinking it is like first question but it was different. I searched for Haar Wavelet Transform on the net and i get some informations. But i could implement it to some point. After getting transform values i could not handle how to arrange data at the step 7, so my script implements less than half of the procedure and unfortunately gets no result. However i included my incomplete code.

```
% % % % % % % Watermark on Wavelet Domain % % % % %% %

% % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % PREPROCESSING % % %  %% % % % % %
% % % % % % % % % % % % % % % % % % % % % % % % %
file_name='son.wav';
[cover_audio,Fs]=audioread(file_name);
file_name='wm.bmp';
message=imread(file_name);

% Represent the grey-scale image watermark as a two dimensional MxN matrix
message=double(message);
message=round(message./256);
message=uint8(message);

% Read dimensions of the cover audio
Mc=size(cover_audio,1);

% Read dimensions of the message picture
```

```
Mm=size(message,1);
Nm=size(message,2);
message=reshape(message,1,[]);
% Convert an audio signal into byte array.

% Level 1 Haar Decomposition
[A1, H1, L1, D1]=dwt2(cover_audio,'haar');

% level 2 Haar Decomposition
[A2, H2, L2, D2]=dwt2(A1,'haar');

% new=[L2 H2 H1];
% in2= idwt2(A1,H2,V2,D2,'haar');
% final= idwt2(in2,H1,V1,D1,'haar');
```

2.3 Watermark on Frequency Domain

In this question i tried to implement my own algorithm that looks for peak values of fft transform and puts bits accordingly. For the algorithm we need to have bins(blocksize) so as to determine bit embed cycle, also together with this we should select peak frequency.I tried determine these as looking equal loudness curve but i could not understand form the figure and it happened to be different values. As too look in algorithm, first, I determined my bins as 8820 which is 0.2 second because frequency of the wav file is 44100 Hz. I chose peak frequency of watermark embedding as 20 Hz('wm_peak') , 'amply' is the amplification of the watermark.

After preprocessing i scaled '.bmp' file as '.wav' file and put in a array. Then in the for loop ,as increasingly size of the block determined, i first take fft of sound then i look different peak values to encode the figure bits. Bit is embedded if fft peaks is lower than the peak i determined and with amplification 'amply' determined. After that i take inverse fft of it and write the file.

I get quite good sound with very little noise because i got fairly good SNR value.

- I give 4 points for the MOS value.
- My SNR value was 15.07.

```
% % % % % % % Watermark on Frequency Domain % % % % %% %

% input and output file names for watermarking

audio_input='son.wav';
watermark_input='wm.bmp';
output_marked='freqwatermarked.wav';
wm_output='wm_ex.bmp';

blocksize=8820;  % For a track with 44100 samples per second
                 % this block size corresponds to a duration of 0.2 s
                 % therefore we have a fft frequency resolution of 5 Hz.
first_block=1;  % to leave about 6 s empty
wm_peak=20;      % corresponds to a low frequency peak for the embedding;

amply=7;       % amplification factor for the watermarked spectral value

% Read audio-file, determine available watermark size
[data,Fs]=audioread(audio_input);
data_orig=data; %hold value to calculate snr value
M=size(data,1);
N=floor(M/blocksize);  % both channels are used for watermark embedding
```

```
23
24 % Watermark preparation
25
26 wm1=imread(watermark_input);  % read watermark image
27 z1=size(wm1,1);
28 z2=size(wm1,2);
29 scale=1;
30 if(N<(z1*z2))
31     scale=N/(z1*z2);
32 end
33 wm_temp=mat2gray(wm1);
34 if(scale~=1)
35     wm_temp=imresize(wm_temp,scale);
36 % possibly scale watermark image to fit the audio file length
37 end
38
39 s1=floor(size(wm_temp,1)/2)*2;       % an even number
40 s2=floor(size(wm_temp,2)/2)*2;
41 wm=imresize(wm_temp,[s1 s2]);   % new image with even side lengths
42 wm_length=s1*s2;                % an even wm_length
43 wm_vec=round(reshape(wm,wm_length,1)./256);
44 if(wm_length>(N-first_block+1))
45     error('Error: Watermark to long');
46 end
47 wm_pad(1:length(wm_vec))=wm_vec; % watermark bytes in a vector
48
49 % Watermark embedding
50 l=-1;
51 for k=first_block:wm_length/2+first_block-1   % k runs over the blocks
52     l=l+2;            % l runs through the watermark bytes in steps by 2
53     x=data((k-1)*blocksize+1:k*blocksize,1);
54     y=fft(x);
55     mark=0;       % search peaks with different absolute values
56                   % beginning at number wm_peak
57     n=wm_peak;    % start fft peak number
58       while(abs(y(n))==abs(y(n+1)) && mark==0)
59          n=n+1;
60          if(n==floor(blocksize/2))
61             mark=1;
62          end
63       end
64     if(n~=floor(blocksize/2))   % otherwise do nothing, i.e.
65                                 % then the watermark byte will be white
66                                 % for this block in the detection
67                                 % algorithm
68       if(wm_pad(l)==0)      % Encode |y(n)|>|y(n+1|
69
70             if(abs(y(n))<abs(y(n+1)))% change the values, if necessary
71                 temp1=y(n+1);
72                 y(n+1)=y(n);
73                 y(n)=temp1;
74             end
75             y(n)=y(n)*amply;       % an amplifying for better detection
76             y(blocksize-n+2)=conj(y(n));
77             y(blocksize-n+1)=conj(y(n+1));
78
79       end
80       elseif(wm_pad(l)==1)         % white is embedded iff
81           if(abs(y(n))>abs(y(n+1)))    % [ |y(n)|<|y(n+1)| or all
82                 temp1=y(n+1);          % fft peaks have equal
83                 y(n+1)=y(n);           % absolute values ]
84                 y(n)=temp1;
```

```
85            end
86              y(n+1)=y(n+1)*amply;              %
87              y(blocksize-n+2)=conj(y(n));      % analogously to above
88              y(blocksize-n+1)=conj(y(n+1));    %
89
90       end
91      % inverse transform of the watermarked fft block
92      data((k-1)*blocksize+1:k*blocksize,1)=ifft(y);
93  end
94
95
96  % Write watermarked audio-file
97  audiowrite(output_marked,data,Fs);
98
99  %%%%%%%%%%%%%%%%%%%%%%%%%%%
100 %%%%%%%% SNR Value%%%%%%%%
101 %%%%%%%%%%%%%%%%%%%%%%%%%%%
102
103 s = mean(data_orig.^2)/mean((data_orig-data).^2);
104 snr = 10.^log(s);
```