FALL-SEM 2021-22: PROJECT REPORT

School of Electronics Engineering (SENSE)

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

ECE2006 DIGITAL SIGNAL PROCESSING (Slot: G2)

Audio Watermarking and Speech Enhancement using Kalman Filter

SUBMITTED TO:

Prof. Rajesh Neelakandan

Associate Professor (Senior)

School of Electronics Engineering

SUBMITTED BY:

ASHWIN V PILLAI – 19BEC0241

A KEVIN GEORGE – 19BEC0807

J RAJA CHIDAMBARAM - 19BEC0811

V SATHIYAPRIYA – 19BEC0815

Abstract:

In this project, we have realized the importance of watermarking a signal, which enables copyright protection and verification. Therefore, we are going to add a watermark to the audio signal using MATLAB.

Noise removal is very important because noise corrupts the speech and causes severe difficulties in various communication environments. Degradation of speech severely affects the ability of a person, whether impaired or normal hearing, to understand what the speaker is saying. For enhancing the watermarked audio, we use a Kalman filter. Kalman filters are used to optimally estimate the variables of interests when they can't be measured directly, but an indirect measurement is available. They are used to find the best estimate of states by combining measurements from various sensors in the presence of noise.

So, summing up, we first watermark the audio signal and in order to enhance the watermarked audio signal, we used a Kalman filter. We are going to implement these using the MATLAB software.

Novelty:

Watermarking will help us in:

- Copyright protection: The copyright owner will be authenticated by the knowledge of the secret key to read the secret watermark.
- Monitoring Embedding: a secret watermark to enable the tracing of illegal copying.
- Fingerprinting: In point-to-point distribution environments information about authenticated customers can be embedded as secret watermarks right before the secure delivery of the data.
- Indication of content manipulation: The indication of content manipulation (tamper-proofing) from the authorized state could be detected by means of a public.

Kalman filter is different from other filters in the following ways:

- Using a windowed Kalman filter for re-linearization past states or when having correlated observations thru time steps, it is often much easier to use the normal equations.
- In addition, the covariance matrix of the Kalman filter can run into non positive semi definiteness over time.
- They have the advantage that they are light on memory (they don't need to keep any history other than the previous state), and they are very fast, making them well suited for real time problems and embedded systems.

Components Required:

- MATLAB.
- Audio file
- Image to be watermarked.

Watermarking:

Watermarking is the process of embedding hidden information into a carrier signal or any piece of digital data. Here the hidden information is also a form of digital data which can be an image, text, audio. Etc. This hidden information is typically used to identify the ownership of the digital data that is watermarked.

Two types of watermarking:

Fig: A visible watermark on top of image.

- Traditional watermarking:

    It is applied to visible media such as images or videos, in this type watermarking the watermark is visible to the naked eye.

- Digital watermarking:

    Here the carrier signal may be audio, images, videos, 3-D models, texts. The watermark is invisible to the naked eye and secretly embedded into the signal without changing the properties of the signal drastically using certain algorithms.

Watermarking using LSB algorithm:

Every n-bit binary value, has a Least Significant Bit (LSB) and a Most Significant Bit (MSB). In LSB algorithm we overwrite the LSB with a bit from the watermark data to secretly embed data into the carrier without drastically affecting its properties.

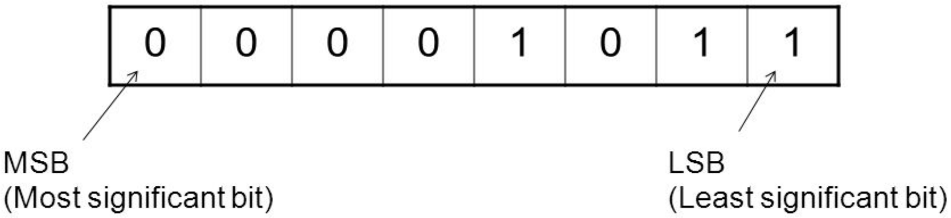


Fig: Depicts the lsb and msb of a 8-bit binary value.

Advantages:

- It is a simple method.
- It can survive transformations like cropping, undesirable noise or compression.
- Security will be enhanced.

Disadvantages:

- A more sophisticated attack that could simply set the LSBs of each pixel to 1 can fully defeat the watermark with a negligible impact on the cover object.
- This way, the embedded watermark can be modified by the attacker

LSB algorithm workflow:

We split this process into two sections, first is embedding the watermark into the host and second section is the retrieval of watermark from a processed signal.

Insertion of watermark:

- First thing we need to do is read the audio signal using ‘*audioread’* function available in MATLAB. This function returns a one- dimensional array of audio samples which ranges from -1 to +1. It also returns the sampling frequency.

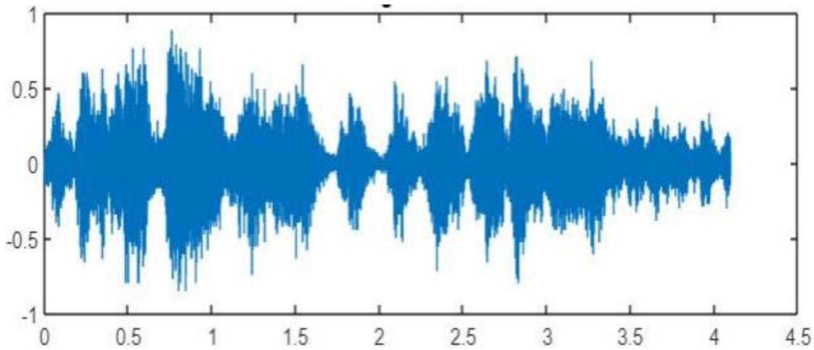


Fig: The sample audio signal we used for the project, its sampled at 8 khz to give 32768 samples.

- Next, we read the image to be watermarked using *‘imread’* function. This function returns a mxn array of data representing each pixel with values from 0 to 255. We make sure it’s a gray scale image for easier implementation.

	1	2	3	4	5	6	7	8	9	10
1	2	0	0	6	0	0	0	1	1	0
2	0	254	255	253	252	252	255	254	253	6
3	0	255	0	0	6	0	0	0	255	0
4	3	255	7	0	0	255	254	1	254	3
5	3	250	0	4	0	1	255	0	255	0
6	0	255	4	255	0	1	0	2	255	8
7	2	255	0	250	255	0	6	0	250	0
8	0	255	0	2	0	3	0	4	255	0
9	1	252	255	253	255	255	255	253	254	3
10	0	5	0	0	0	2	0	4	2	0

Fig: The image(10x10) we have taken for watermarking and its corresponding pixel values in the form of matrix.

- We then convert the host array to range from values of 0 to 255 to match the range of pixel values. We do this by using ‘*uint8’* function. Once we convert it to values ranging from 0 to 255, we again convert it to binary format to give us an array of 8-bit binary values. Hence, we convert the array from ‘-1 to +1’ to ‘0 to 255’ to ‘8-bit array’.

32768×1 double

	1
1	-0.0330
2	-0.0096
3	0.0295
4	0.0363
5	0.0121
6	-0.0070
7	-0.0009
8	0.0329
9	0.0449
10	0.0449
11	0.0244
12	-0.0004
13	-0.0313
14	-0.0449
15	-0.0313
16	-0.0313
17	-0.0296
18	-0.0279

32768×1 uint8

	1
1	124
2	127
3	132
4	133
5	130
6	127
7	128
8	132
9	134
10	134
11	131
12	128
13	124
14	122
15	124
16	124
17	124
18	124

32768×8 char

```
01111100
01111111
10000100
10000101
10000010
01111111
10000000
10000100
10000110
10000110
10000011
10000000
01111100
01111010
01111100
01111100
01111100
01111100
01111110
10000001
10000011
10000011
10000010
10000100
10000101
10001000
10001000
10000110
```

Fig: The 32768 samples we receive from audio file are converted consequently to obtain its 8-bit binary value.

- Next step is to also convert the image matrix consisting of pixel values to a one-dimensional array consisting of 8-bit binary values using '*dec2bin'* function.

	1	2	3	4	5	6	7	8	9	10
1	2	0	0	6	0	0	0	1	1	0
2	0	254	255	253	252	252	255	254	253	6
3	0	255	0	0	6	0	0	0	255	0
4	3	255	7	0	0	255	254	1	254	3
5	3	250	0	4	0	1	255	0	255	0
6	0	255	4	255	0	1	0	2	255	8
7	2	255	0	250	255	0	6	0	250	0
8	0	255	0	2	0	3	0	4	255	0
9	1	252	255	253	255	255	255	253	254	3
10	0	5	0	0	0	2	0	4	2	0

100×8 char

```
00000010
00000000
00000000
00000011
00000011
00000000
00000010
00000000
00000001
00000000
00000000
11111110
11111111
11111111
11111010
11111111
11111111
11111111
11111100
00000101
00000000
11111111
00000000
00000111
00000000
00000100
00000000
```

Fig: The 100 pixels in our 10x10 image is also converted consequently to obtain its 8-bit binary value in the form of an array.

- Next, we take each bit from our image array and replace the lsb of a sample with this image bit. In our case, since our image is 10x10 image, we get 100 pixels and each pixel is represented by a 8-bit binary value, hence we get total of 800 bits and each bit replaces lsb of one sample. Therefore, 800 samples get their lsb replaced. To sum up, 8 samples in order contains information of 1 pixel of the image used for watermarking. Also the number of image bits has to be greater than or equal to the number of samples in order to embed the complete image into the audio file.

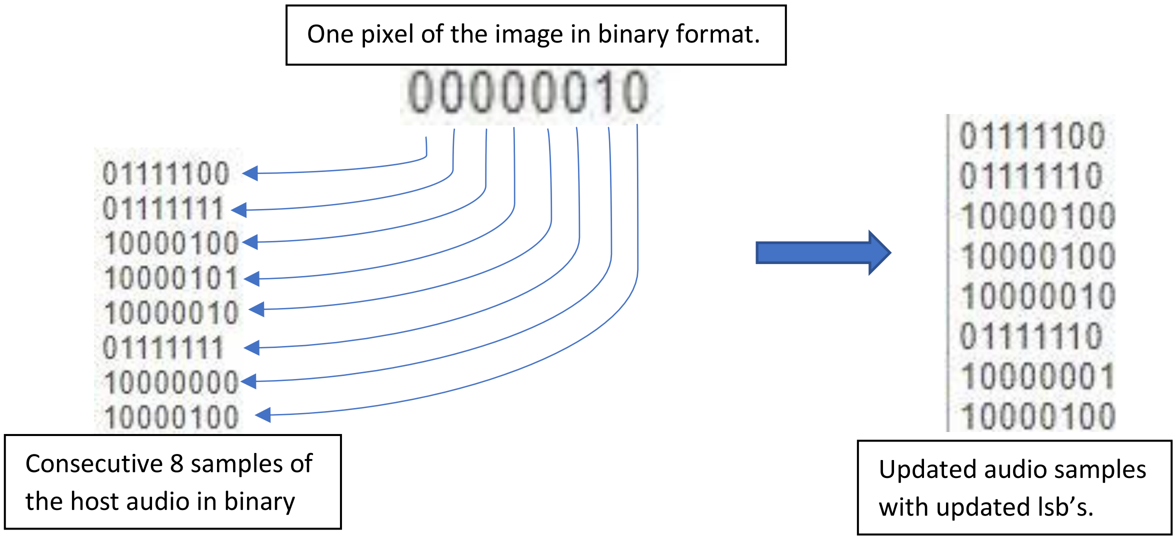


Fig: Figure shows how the lsb's of each sample are updated to accommodate the information of the image.

- Once, we follow this procedure until all the bits are embedded. Once, we complete the new audio file is plotted and used as our watermarked audio using *'audiowrite'* function.

Retrieval of watermark for verification:

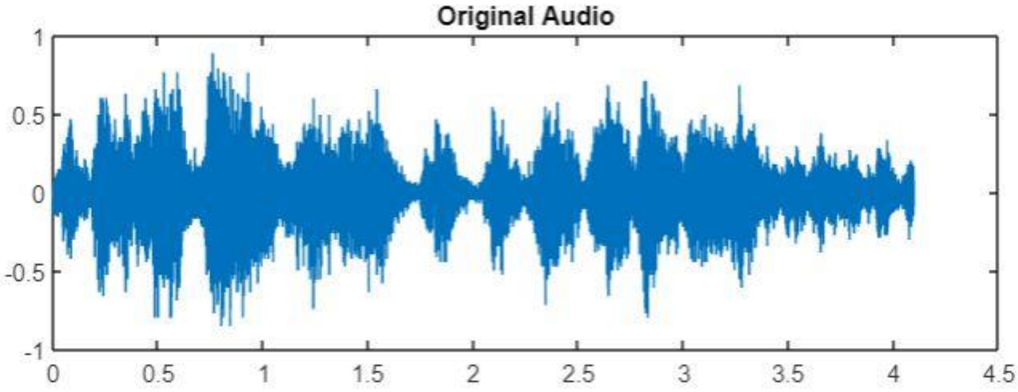
- Now, we can verify a audio signal to identify ownership by checking if a certain audio file contains the watermark or not by using the same algorithm. The image retrieved can be checked with our original watermark image, if it matches then the signal is copyrighted.
- For this, we do the reverse process of the above-mentioned workflow. First, we take a sample audio file and convert it to an 8-bit binary array.
- Then we extract lsb's from each sample and create 8-bit binary values which represent pixel value.
- Once, we receive desired pixels, we convert it into decimal format using *'bin2dec'* format. These are then compiled into a mxn matrix. This matrix is then plotted as an image using *'imshow'* function.
- The received image can now be compared with our original watermark and can be thoroughly analysed.

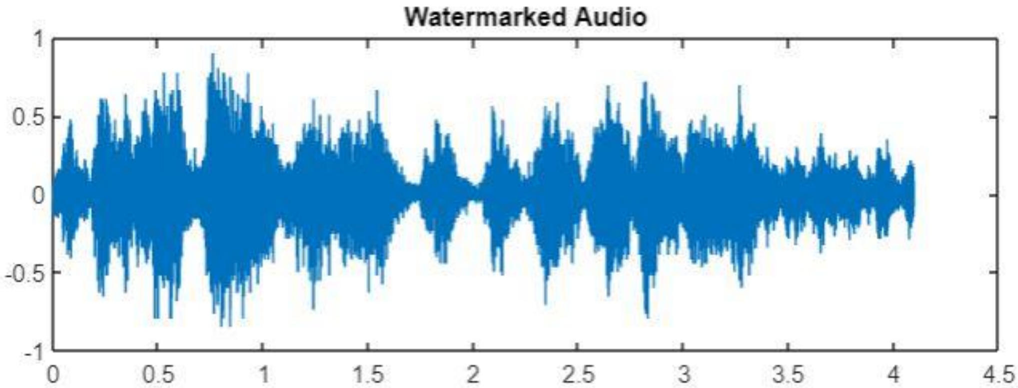
MATLAB code for inserting watermark:

```
clc;
clear;
close all;
[host, f] = audioread('Laughter-16-8-mono-4secs.wav');
dt=1/f;
t =0:dt:(length(host)*dt)-dt;

subplot(2,1,1)
plot(t,host)
title('Original Audio')

host = uint8(128*(host + 1));
wm = imread('image.jpeg');
wm=rgb2gray(wm);

[r, c] = size(wm);
wm_l = length(wm(:))*8;

if length(host) < wm_l
    disp('your image pixel is not enough')
else
    host_bin = dec2bin(host, 8);
    new_host_bin=dec2bin(host, 8);
    wm_bin = dec2bin(wm(:), 8);
    wm_str=zeros(wm_l,1);
    for j = 1:length(wm(:))
        for i = 1:8
            ind = (j-1)*8 + i;
            wm_str(ind) = str2double(wm_bin(j, i));
        end
    end
    for i = 1:wm_l
        host_bin(i, 8) = dec2bin(wm_str(i));
    end
    host_n = bin2dec(host_bin);
    host_new = 2*(double(host_n)/255 - 0.5);

    subplot(2,1,2)
    plot(t,host_new)
    title('Watermarked Audio')
    figure;

    audiowrite('host_new.wav',host_new, f);
    %soundsc(host_new, f);
end
```

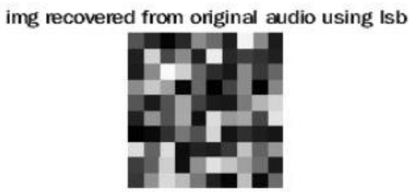
```matlab
%% recovery of image from Watermarked audio %%
subplot(3,1,1)
imshow(wm);
title('Original img')

wmr = dec2bin((host_new/2 + 0.5)*255);
wmr_str = zeros(wm_l,1);
for i = 1:wm_l
    wmr_str(i)=bin2dec(wmr(i,8));
end
wmr_img=dec2bin(zeros(length(wm(:)),1),8);
for j = 1:length(wm(:))
    for i = 1:8
        ind = (j-1)*8 + i;
        wmr_img(j, i) = dec2bin(wmr_str(ind));
    end
end
wmr_img = bin2dec(wmr_img);

wmr_final=zeros(r,c);
for j = 1:c
    for i = 1:r
        ind = (j-1)*r + i;
        wmr_final(i, j) = wmr_img(ind,1);
    end
end
subplot(3,1,2)
imshow(uint8(wmr_final));
title('img recovered from watermarked audio using lsb')
%% recovery of an image from original audio
wmn = new_host_bin;
wmn_str = zeros(wm_l,1);
for i = 1:wm_l
    wmn_str(i)=bin2dec(wmn(i,8));
end
wmn_img=dec2bin(zeros(length(wm(:)),1),8);
for j = 1:length(wm(:))
    for i = 1:8
        ind = (j-1)*8 + i;
        wmn_img(j, i) = dec2bin(wmn_str(ind,1));
    end
end
wmn_img = bin2dec(wmn_img);

wmn_final=zeros(r,c);
for j = 1:c
    for i = 1:r
        ind = (j-1)*r + i;
        wmn_final(i, j) = wmn_img(ind,1);
    end
end
subplot(3,1,3)
imshow(uint8(wmn_final));
title('img recovered from original audio using lsb')
```

Output:





From the above graphs its evident that the audio remains unchanged to the naked eye but still inherits data of an image.



We can see that its only possible to retrieve watermark from watermarked audios and not from normal audio signals.

Enhancement of watermarked audio using Kalman Filter:

- The main aim of the work is speech enhancement using Kalman filters. Initially, we have taken an audio signal and then watermarked it using the Least Significant Bit method. Once the signal is watermarked, we enhance it using a Kalman filter.
- The watermarked audio is then verified and is enhanced by removing any noise using the Kalman filter.
- The Kalman filter produces an estimate of the state of the system as an average of the system's predicted state and of the new measurement using a weighted average.
- The purpose of the weights is that values with better (i.e., smaller) estimated uncertainty are "trusted" more. The weights are calculated from the covariance, a measure of the estimated uncertainty of the prediction of the system's state.
- The result of the weighted average is a new state estimate that lies between the predicted and measured state, and has a better estimated uncertainty than either alone.
- This process is repeated at every time step, with the new estimate and its covariance informing the prediction used in the following iteration.
- This means that Kalman filter works recursively and requires only the last "best guess", rather than the entire history, of a system's state to calculate a new state.

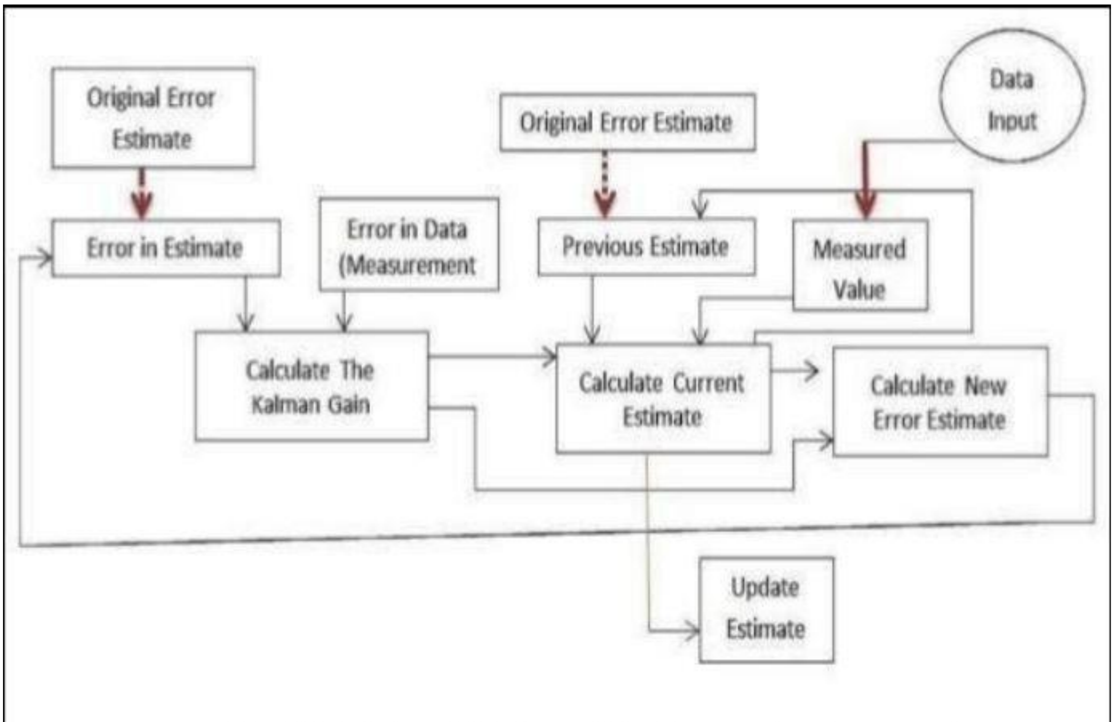
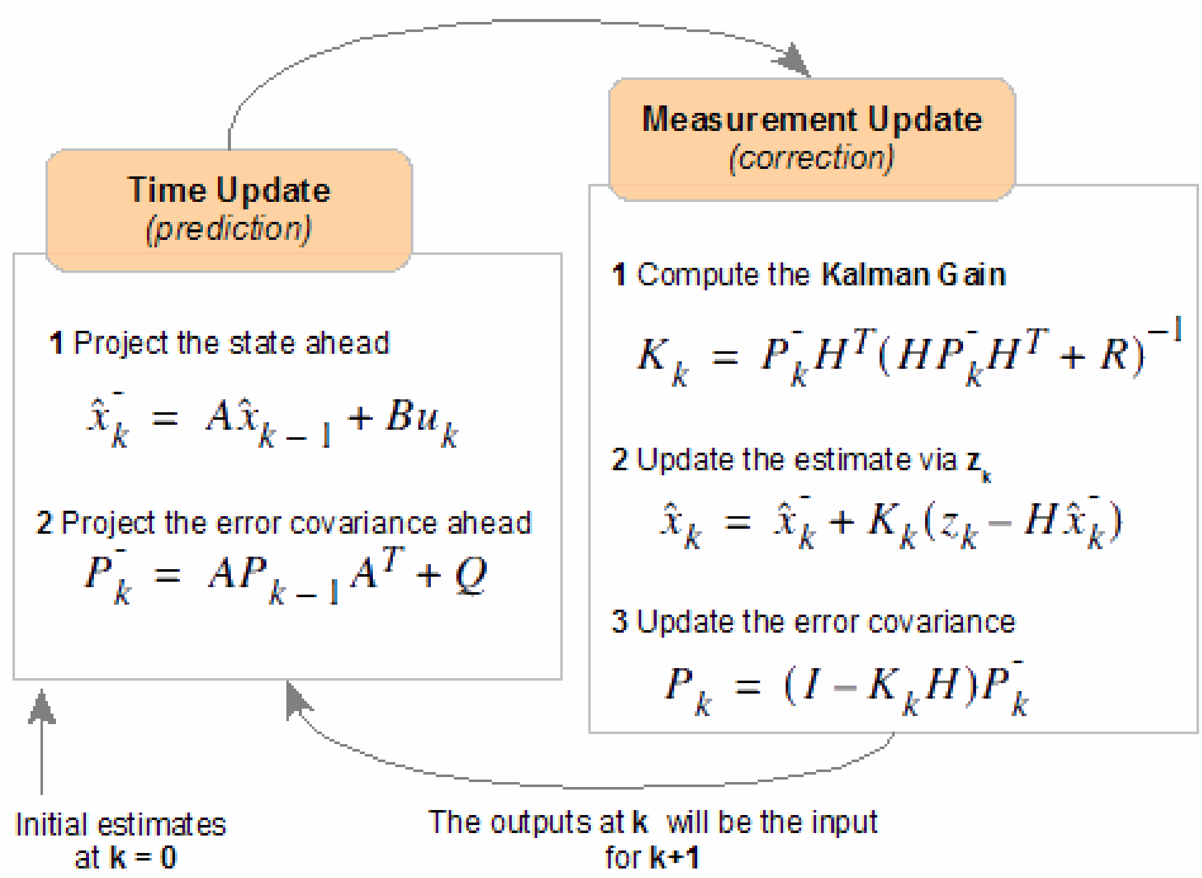


Fig: Mechanism of Kalman filters in speech enhancement

Kalman Filter equations:

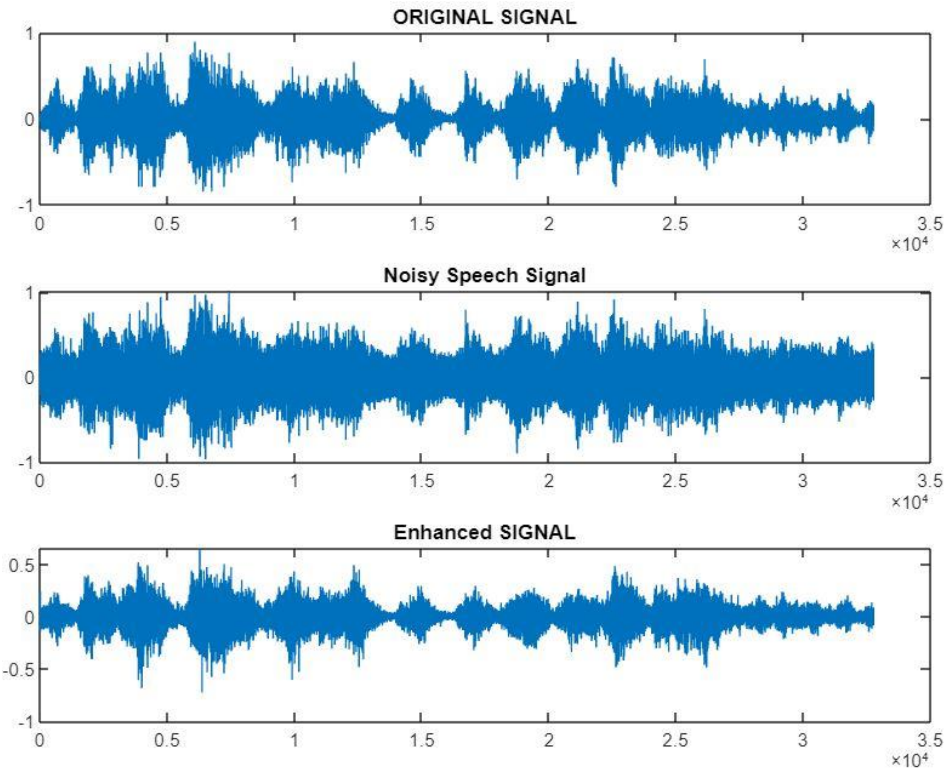


MATLAB code for inserting watermark:

```
clc;
clear all;
close all;
[x,fs]= audioread('host_new.wav');
v=(rand(size(x))-0.5)*0.5; % adding a random Gaussian noise eqn
orig=x+v; % noisy speech signal
no=orig;
N=length(x); % length of the input signal
F = zeros(5, N); % initialization of standard transition matrix
I = eye(5); % transition matrix
H = zeros(5, N);
sig = zeros (5, 5*N); % priori or posteri covariance matrix.
K = zeros (5, N); % kalman gain.
XX = zeros (5, N); % kalman coefficient for yy.
y = zeros (1, N); % requiring signal (desired signal)
vv = zeros (1, N); % predicted state error vector
yy = zeros (1, N); % Estimated error sequence
Q = 0.0001*eye (5, 5); % Process Noise Covariance.
R = 0.1; % Measurement Noise Covariance
y=x (1: N); % y is the output signal produced.
sig (1:5, 1:5) = 0.1*I;
for k=6: N
    F(1:5,k)=-[y(k-1);y(k-2);y(k-3);y(k-4);y(k-5)];
    H(1:5,k)=-[yy(k-1);yy(k-2);yy(k-3);yy(k-4);yy(k-5)];
```

```
    K(1:5,k)=sig(1:5,5*k-29:5*k-25)*F(1:5,k)*inv(F(1:5,k)'*sig(1:5,5*k-
29:5*k-25)*F(1:5,k)+R); %Kalman Gain
    % error covariance matrix
    sig(1:5,5*k-24:5*k-20)=sig(1:5,5*k-29:5*k-25)-sig(1:5,5*k-29:5*k-
25)*F(1:5,k)*inv(F(1:5,k)'*sig(1:5,5*k-29:5*k-25)*F(1:5,k)
+R)*(F(1:5,k)'*sig(1:5,5*k-29:5*k-25))+Q;
    XX(1:5,k) =(I - K(1:5,k)*F(1:5,k)')*XX(1:5,k-1) + (K(1:5,k)*y(k)); %
posteriori value of estimate X(k)
    orig (k) =y (k)-(F (1:3, k)'*XX (1:3, k)); % estimated speech signal
    yy (k) = (H (1:5, k)'*XX (1:5, k)) + orig (k); % no. of coefficients
per iteration
end
tt = 1:1: length(x);
figure (1);
subplot (311);
plot(x);
title ('ORIGINAL SIGNAL');
subplot (312);
plot(no);
title ('Noisy Speech Signal');
subplot (313);
plot(orig);
title ('ESTIMATED SIGNAL');
figure (2);
plot(tt, no,'r');
hold on;
plot(tt, orig,'b');
title ('Combined plot');
legend ('Noise', 'estimated');
```

Output:



Here, we can have added noise to the watermarked audio and then enhanced it using Kalman equations via MATLAB.

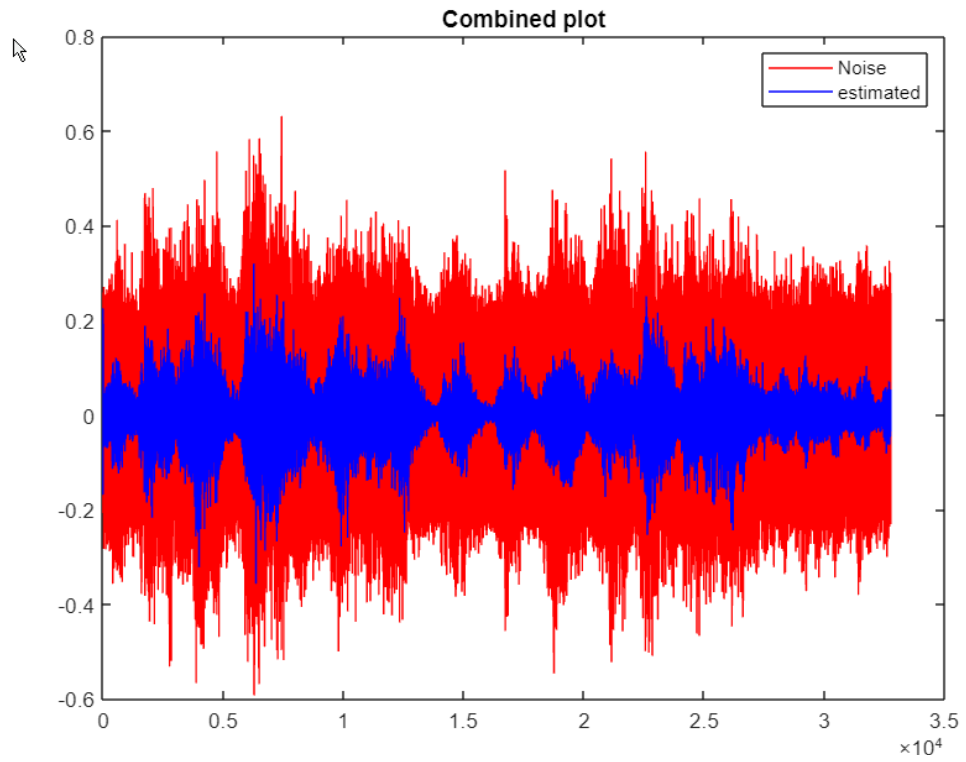


Figure shows the combined plot of enhanced audio and the watermarked audio with added noise.

Future enhancements and alternatives:

Our project is limited to watermarking of audio currently, we can extend the applicability to images, videos, texts and much more. We can also use other alternatives to the LSB algorithm such as:

- Pixel value differencing (PVD):
- Edges based data embedding method (EBE)
- Random pixel embedding method (RPE)
- Labelling or connectivity method
- BPCP method.

With more complex alternatives we can create a more secure watermarked audio. Also, we can improve our enhancement techniques without disturbing the watermark data by using better alternatives. Few alternatives to Kalman filter are:

- Spectral Subtraction approach
- Wiener filter
- Weighted filter.

With slight improvement, the project can be used in various industries for copyright protection, mainly in the music industry which is vulnerable to around 30% of piracy attacks.

Conclusions:

For watermarking, we used the LSB technique, because it has little effect on the signal, and is a very simple and straight-forward method. It can easily survive transformations like cropping, undesirable noise or compression. For speech enhancement, we have used a Kalman filter, because it is time-domain in nature. In our daily lives, the signals are not stationary and will vary randomly. Hence, Kalman filter is suitable for both stationary and non-stationary signal.

References:

- Nejad, M. Y., Mosleh, M., & Heikalabad, S. R. (2019). An LSB-based quantum audio watermarking using MSB as arbiter. *International Journal of Theoretical Physics*, *58*(11), 3828-3851.
- Xiong, Y. (2006, November). Covert communication audio watermarking algorithm based on LSB. In *2006 International Conference on Communication Technology* (pp. 1-4). IEEE.
- Subbarayan, S., & Ramanathan, S. K. (2009, December). Effective watermarking of digital audio and image using Matlab technique. In *2009 Second International Conference on Machine Vision* (pp. 317-319). IEEE.
- Gannot, S., & Moonen, M. (2003, September). On the application of the unscented Kalman filter to speech processing. In *IEEE Int. Workshop on Acoustic Echo and Noise Control (IWAENC), Kyoto, Japan* (p. 811).
- Paliwal, K., & Basu, A. (1987, April). A speech enhancement method based on Kalman filtering. In *ICASSP'87. IEEE International Conference on Acoustics, Speech, and Signal Processing* (Vol. 12, pp. 177-180). IEEE.
- Goh, Z., Tan, K. C., & Tan, B. T. G. (1999). Kalman-filtering speech enhancement method based on a voiced-unvoiced speech model. *IEEE Transactions on speech and audio processing*, *7*(5), 510-524.
- Mr. Jayesh Surana, Aniruddh Sonsale, Bhavesh Joshi, Deepesh Sharma (2017), Steganography Techniques. 2017 IJEDR | Volume 5, Issue 2 | ISSN: 2321-9939