ECE2006 – Digital Signal Processing

Project Report

Project Title: Digital video watermarking Using MATLAB

SLOT: G1

Team Members:

Karthik

17BEC0695

Submitted to:

Prof.
Prakasam P

ABSTRACT

Due to increase in growth of internet users of networks are increasing rapidly. Owners of the digital products are concerned about illegal copying of their products. Security and copyright protection are becoming important issues in multimedia applications and services. Digital watermarking is a technology used for copyright protection of digital media. Here ownership information data called watermark is embedded into the digital media without affecting its perceptual quality. In case of any dispute, the watermark data can be detected or extracted from the media and use as a proof of ownership. Digital video watermarking scheme based on Discrete Wavelet Transform is addressed in this paper. Design of this scheme using Matlab is proposed. Embedded watermark is robust against various attacks that can be carried out on the watermarked video

INTRODUCTION

The use of digital multimedia content is increased large amount of data is transfer and distributed easily. This development will benefit multimedia product owners as sales will increase. Also it will pose challenge to their ownership as most of multimedia products are distributed in insecure format. These products can be transmitted and redistributed easily without any authentication as various tools are available at no cost. So there is need for copyright protection of multimedia data. Video become an important tool for the entertainment and educational industry. Digital video watermarking is new technology used for copyright protection of digital media. It inserts authentication information in multimedia data which can be used as proof of ownership.

The watermarking technique is used for data hiding. Video watermarking algorithms normally prefers robustness. Most of the proposed video watermarking schemes are based on the techniques of image watermarking. But video watermarking introduces some issues not present in image watermarking. Watermarking techniques

can be classified into spatial or frequency domain by place of application. Spatial domain watermarking is performed by modifying values of pixel colour samples of a video frame whereas watermarks of frequency domain techniques are applied to coefficients obtained as the result of a frequency transform of either a whole frame or single block-shaped regions of a frame.

Most commonly used transforms are

1. Discrete Fourier Transform (DFT),

2. Discrete Cosine Transform (DCT),

3. Discrete Wavelet Transform (DWT).

Video Watermarking

Maximum occurrences of copyright violation and distribution happen for video media content. So Video Watermarking is one of the most accepted techniques among the various Watermarking techniques currently in use.

Requirements for video water marking

Requirements for video Watermarking are as follows:

1. Video data is subject to increased attacks than any other media.

2. Video content is sensitive to distortions and Watermarking may degrade the quality.

3. Video compression algorithms are computationally rigorous.

4. Video require large bandwidth that is why it is mostly carried in compressed domain. So Watermarking algorithms also adaptable for compress area processing.

METHODOLOGY

Discrete Wavelet Transform (DWT)

The Discrete Wavelet Transform (DWT) is used in a wide variety of signal processing applications. 2-D discrete wavelet transform (DWT) decomposes an image or a video frame into sub-images, 3 details and 1 approximation. The 2-D DWT is an application of the 1-D DWT in both the horizontal and the vertical directions. DWT separates the frequency band of an image into a lower resolution approximation sub-band (LL) as well as horizontal (HL), vertical (LH) and diagonal (HH) detail components.

Watermark is embedded in low frequencies obtained by Wavelet decomposition which increases the robustness. So that resultant watermark video become susceptible to different attacks that have low pass characteristics like filtering, lossy compression and geometric distortions.

Watermark embedding process

The Watermark embedding process consists of the following steps:

1. Video is divided into frames RGB frames are converted to YUV frames.

2. 2-DWT is applied on it.

3. RGB watermark image is converted into a vector P= {p1, p2… p32x32} of zeros and ones.

4. This vector P is again divided into n parts. Then each part is embedded into each of the corresponding LL and HH sub bands. The watermark pixels are embedded with strength x into the maximum coefficient Mi of each PC block Yi. The embedding equation is:

$$Mi=Mi+ xW$$

Where, x is the watermark embedding strength.

5. Inverse DWT is applied to obtain the watermarked luminance component of the frame. Finally, watermarked frame is reconstructed and watermarked video is obtained.

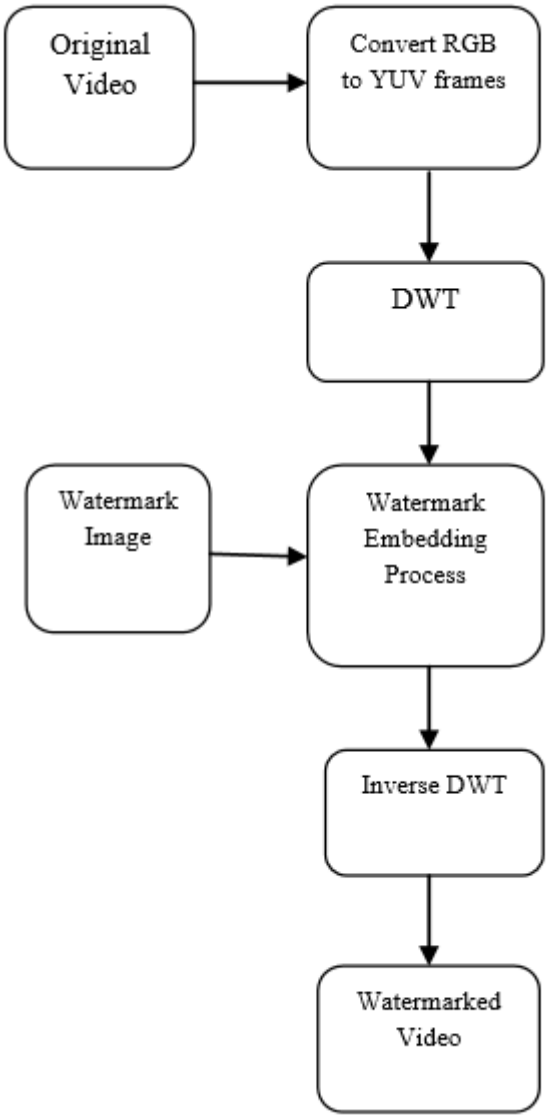


Fig: Watermarking embed process

Watermark Extraction Process

The steps used for watermark extraction is the same as the steps in the embedding but in the reverse direction. As follows

1. Watermarked video is converted into frames. Each RGB frame is converted to YUV representation.

2. DWT is applied. LL and HH sub-bands divided into nxn non-overlapping blocks.

3. Following equation is used to extract watermark

$$W = \frac{M_i^1 - M_i}{x}$$

4. The extracted watermark is compared with the original watermark as follows:

$$NC = \frac{\sum_i \sum_j W(i,j).W'(i,j)}{\sum_i \sum_j W(i,j)^2}($$

Where, NC is the normalized correlation. NC value is 1 when the watermark and the extracted watermark are identical and zero if the two are different from each other.

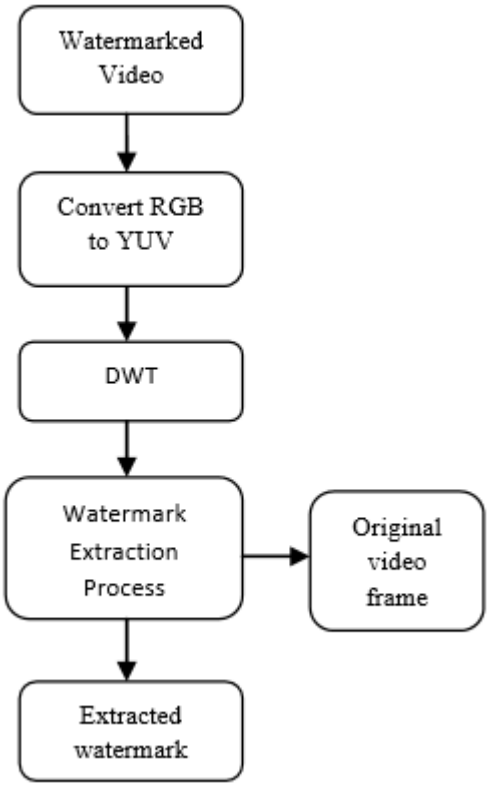


Fig: watermark extraction process

MATLAB CODE:

```
clear all; close all; clc;              % Initialize
vidname = 'C:\Users\Karthik\Desktop\video.mpg';
secimage = 'C:\Users\Karthik\Desktop\secret.png';
% Secret image path
x = 0.1;                                  % Embedding
Strength
% Increase or decrease the value of x to improve or
weaken the embedding
% strength. Higher x values will be more tamper proof,
however, will result
% in a larger loss in SNR.

display('Loading Video...')
tic

vidObj =
VideoReader('C:\Users\Karthik\Desktop\video.mpg');  %
Load video object
```

```
% Video parameters
nFrames = vidObj.NumberOfFrames;
vidH = vidObj.Height;
vidW = vidObj.Width;

% Create empty frames
mov(1:nFrames) =
struct('cdata',zeros(vidH,vidW,3,'uint8'),'colormap',[]);

% Copy video frames into mov struct
for k = 1:nFrames
    mov(k).cdata = read(vidObj,k);
end
display(['Loaded ' num2str(nFrames) ' frames
successfully...']);
toc

%% Frame by Frame Embedding
display('Begin Embedding...')
tic

secret = imread(secimage);
% convert image into binary (0.1 is a predetermined
threshold)
S = im2bw(secret,0.1);
[wmh,wmw] = size(S);
RGB(1:nFrames) =
struct('cdata',zeros(vidH,vidW,3,'uint8'),'colormap',[]);
Yorg = zeros(vidH,vidW,nFrames);
Yf_out = zeros(vidH,vidW,nFrames);
for i = 1:nFrames
    YUV = convRGBYUV(mov(i).cdata);
    Yorg(:,:,i) = YUV(:,:,1);
    % Embedding is done on the Y frame only
    % is converted into grayscale. This makes no sense!
Given the motivation
    % the movie into grayscale makes no sense at all!...
    Yf_out(:,:,i) = dwt_embedd(Yorg(:,:,i),S,x);

    % Convert embedded frame back into RGB (for viewing
purposes)
    RGB(i).cdata =
uint8(convYUVRGB(cat(3,Yf_out(:,:,i),YUV(:,:,2),YUV(:,:,3
))));
end
```

```
display(['Embedded ' num2str(nFrames) ' frames
successfully...']);
toc

%% Frame by Frame Extraction
display('Begin Extracting...')
tic

wMark(1:nFrames) =
struct('cdata',zeros(wmh,wmw,1,'uint8'),'colormap',[]);
for i = 1:nFrames
    wMark(i).cdata =
dwt_extract(Yf_out(:,:,i),Yorg(:,:,i),x,[wmh wmw]);
end

display(['Extracted ' num2str(nFrames) ' frames
successfully...']);
toc

%% Result Generation
display('Generating results for each frame...')


NC = ones(1,nFrames);
NCden = sum(sum(S.*S));
for i = 1:nFrames
    NCnum = sum(sum(wMark(i).cdata.*S));
    if (NCden~=NCnum)
        NC(i) = NCnum/NCden;
    end
end


% PSNR
% Only the Y frame of the YUV image is considered.
Specific method is not
% defined on the paper. Using the provided algorithm...
MSE = abs((1/(vidH*vidW))*(sum(sum(Yorg - Yf_out))));
PSNR = 10*log((255*255)/MSE);
% Note that PSNR values, in the current configuration,
will be high values.
% Higher PSNR suggests a higher imperceptibility. Which
means that, if x is
% decreased from 10 to 0.1, PSNR will increase from ~118
to ~165.

h1=implay(mov,vidObj.FrameRate);
h2=implay(RGB,vidObj.FrameRate);
```

```
h3=implay(wMark,vidObj.FrameRate);

set(h1.Parent, 'position', [150 100 vidW+10
vidH+10],'Name','Original Video')
set(h2.Parent, 'position', [150+vidW+30 100 vidW+10
vidH+10],'Name','Embedded Video')
set(h3.Parent, 'position', [150+2*vidW+60 100 wmh+10
wmw+10],'Name','Extracted Watermark')

display("PSNR = "PSNR)
display("MSE = "MSE)
display('Use the movie player to watch the results...')
```

DWT embed code

```
function [emFrame] = dwt_embedd(frame,secret,x)
% This function embedds an input frame with the secret
image using DWT
[LL,LH,HL,HH]=dwt2(frame,'haar');

% DWT2 transformation - Level 2
[LL21,LH21,HL21,HH21] = dwt2(LL,'haar');
[LL22,LH22,HL22,HH22] = dwt2(HL,'haar');
[LL23,LH23,HL23,HH23] = dwt2(LH,'haar');
[LL24,LH24,HL24,HH24] = dwt2(HH,'haar');

[h,w,~] = size(LL21);
%% Secret Image Manipulation

SS = reshape(secret,1,numel(secret));
blk_size = h*w;                          % Each block size
init = 0;                              % array location
pointer

W_ims = zeros(h,w,8);

for i = 1:8
    W_ims(:,:,i) = reshape(SS(init+1:i*blk_size),h,w);
```

```
    init = i*blk_size;                    % Change array
pointer
end

newLL21 = LL21 + x*W_ims(:,:,1);
newLL22 = LL22 + x*W_ims(:,:,2);
newLL23 = LL23 + x*W_ims(:,:,3);
newLL24 = LL24 + x*W_ims(:,:,4);
newHH21 = HH21 + x*W_ims(:,:,5);
newHH22 = HH22 + x*W_ims(:,:,6);
newHH23 = HH23 + x*W_ims(:,:,7);
newHH24 = HH24 + x*W_ims(:,:,8);

%% Frame Regeneration iDWT
newLL = idwt2(newLL21,LH21,HL21,newHH21,'haar');
newHL = idwt2(newLL22,LH22,HL22,newHH22,'haar');
newLH = idwt2(newLL23,LH23,HL23,newHH23,'haar');
newHH = idwt2(newLL24,LH24,HL24,newHH24,'haar');

emFrame = idwt2(newLL,newLH,newHL,newHH,'haar');
end
```

DWT extract code

```
function [wMark] = dwt_extract(wframe,frame,x,wmsize)
% This function extracts the DWT watermark when the
watermarked frame
[fLL,fLH,fHL,fHH]=dwt2(frame,'haar');

% DWT2 transformation - Level 2
[fLL21,~,~,fHH21] = dwt2(fLL,'haar');
[fLL22,~,~,fHH22] = dwt2(fHL,'haar');
[fLL23,~,~,fHH23] = dwt2(fLH,'haar');
[fLL24,~,~,fHH24] = dwt2(fHH,'haar');

%% wFrame manipulation - DWT2
% DWT2 transformation - Level 1
[wLL,wLH,wHL,wHH]=dwt2(wframe,'haar');

% DWT2 transformation - Level 2
[wLL21,~,~,wHH21] = dwt2(wLL,'haar');
```

```
[wLL22,~,~,wHH22] = dwt2(wHL,'haar');
[wLL23,~,~,wHH23] = dwt2(wLH,'haar');
[wLL24,~,~,wHH24] = dwt2(wHH,'haar');

%% Watermark Regeneration iDWT
[h,w,~] = size(fLL21);
W_ims = zeros(h,w,8);

W_ims(:,:,1) = (wLL21-fLL21)/x;          % Extraction
algorithm
W_ims(:,:,2) = (wLL22-fLL22)/x;
W_ims(:,:,3) = (wLL23-fLL23)/x;
W_ims(:,:,4) = (wLL24-fLL24)/x;
W_ims(:,:,5) = (wHH21-fHH21)/x;
W_ims(:,:,6) = (wHH22-fHH22)/x;
W_ims(:,:,7) = (wHH23-fHH23)/x;
W_ims(:,:,8) = (wHH24-fHH24)/x;

W = [];
for i = 1:8
    W = [W reshape(W_ims(:,:,i),1,h*w)];
end

wMark = reshape(round(W),wmsize(1),wmsize(2));  %
Rounding W for better accuracy
end
```

CONVERT YUV TO RGB

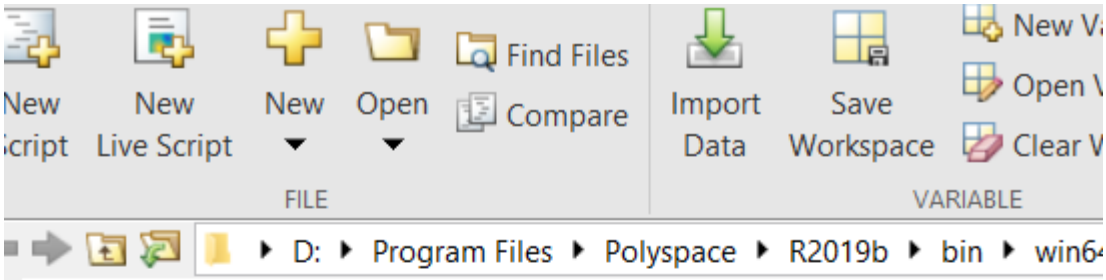
```
function [RGB] = convYUVRGB(YUV)
% Break frames
Y = YUV(:,:,1);
U = YUV(:,:,2);
V = YUV(:,:,3);

% Mapping
R = Y                 +  1.13983 * V;
G = Y - 0.39465 * U -  0.58060 * V;
B = Y + 2.03211 * U                ;

% Concatanate the matrix back
RGB = cat(3,R,G,B);

end
```

CONVERT RGB TO YUV

```
function [YUV] = convRGBYUV(RGB)
% This function converts an RGB frame into YUV
```

```
% Break frames
R = double(RGB(:,:,1));
G = double(RGB(:,:,2));
B = double(RGB(:,:,3));

% Mapping
Y = 0.299 * R + 0.587 * G + 0.114 * B;
U = -0.14713 * R - 0.28886 * G + 0.436 * B;
V = 0.615 * R - 0.51499 * G - 0.10001 * B;

% Concatanate the matrix back
YUV = cat(3,Y,U,V);

End
```

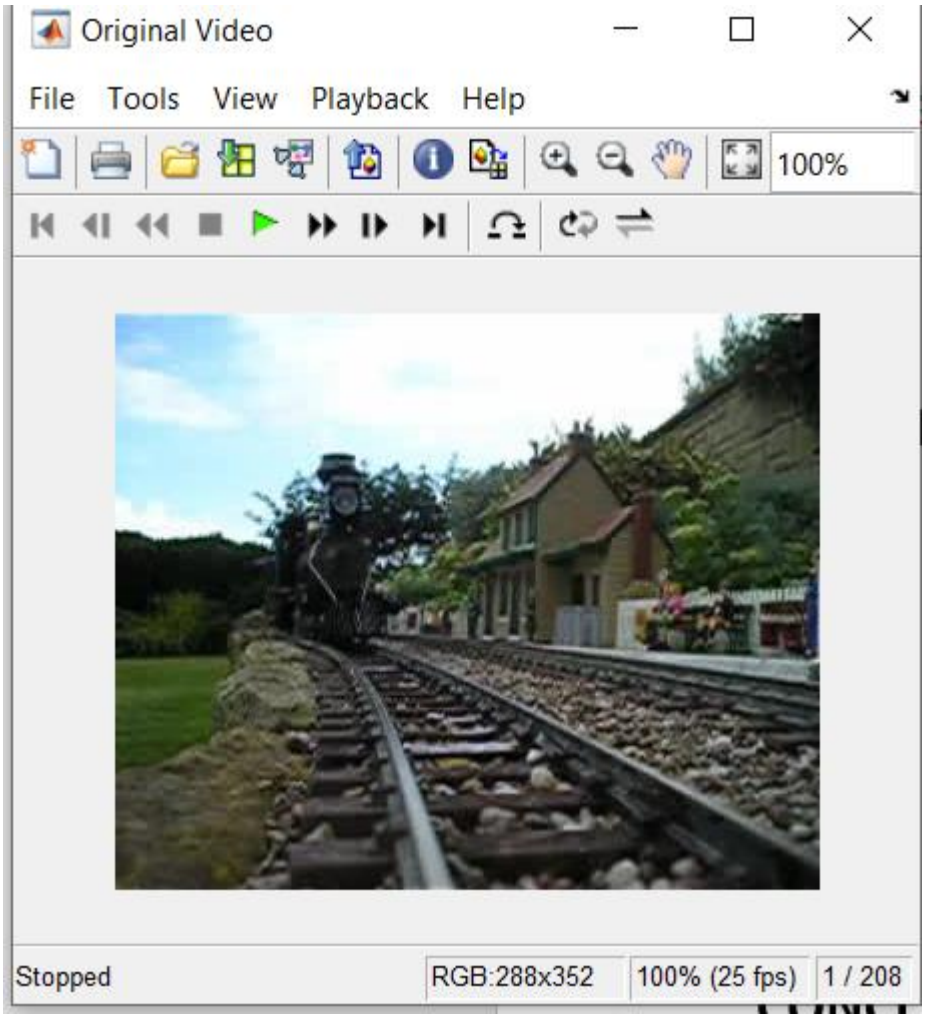
RESULTS



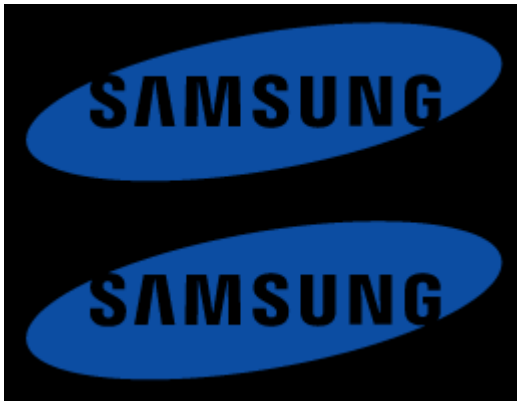

```
Loading Video...
Loaded 208 frames successfully...
Elapsed time is 5.310479 seconds.
Begin Embedding...
Embedded 208 frames successfully...
Elapsed time is 8.919652 seconds.
Begin Extracting...
Extracted 208 frames successfully...
Elapsed time is 4.503439 seconds.
Generating results for each frame...

MSE(:,:,1) =

    0.0045


MSE(:,:,2) =

    0.0045


MSE(:,:,3) =

    0.0045
```

```
MSE(:,:,208) =

    0.0045


PSNR(:,:,1) =

  164.9190


PSNR(:,:,2) =

  164.9190


PSNR(:,:,3) =

  164.9190
```

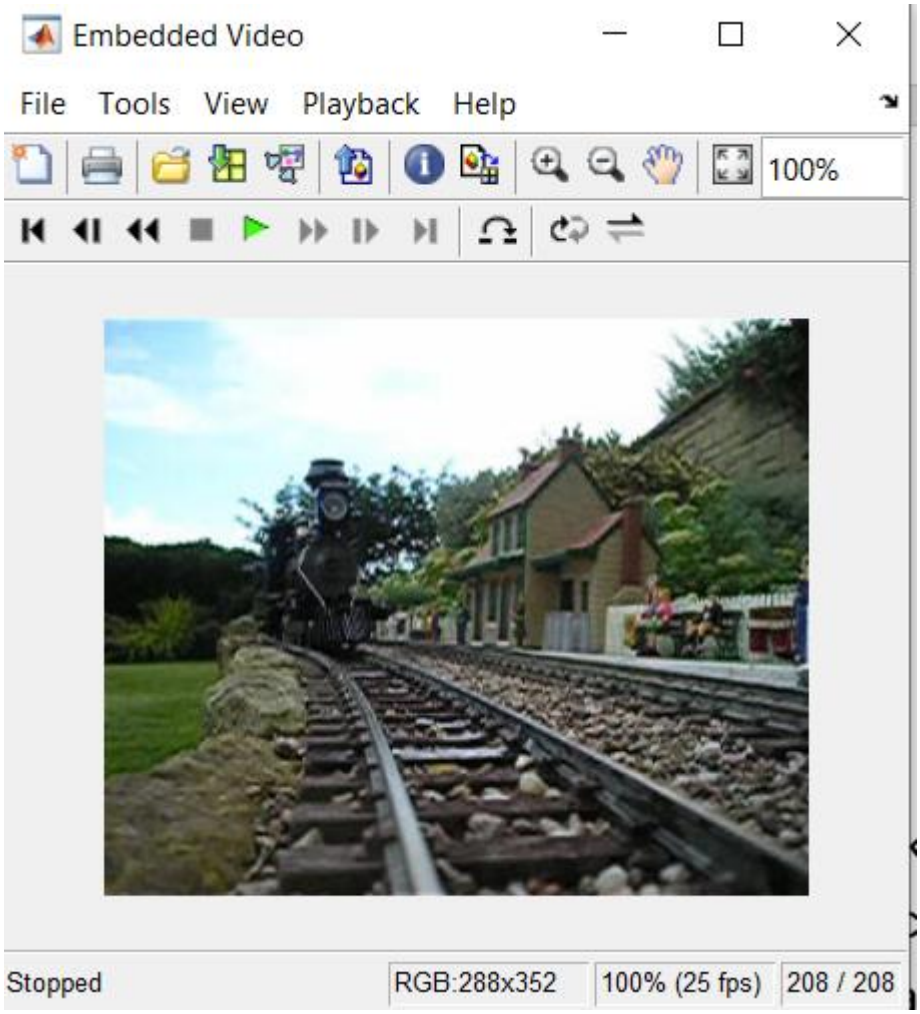
Original video



SECRET IMAGE



EMBEDDED VIDEO



EXTRACTED WATERMARK

CONCLUSION

Here implementation of digital video watermarking scheme based on DWT is proposed. Due to multiresolution characteristics of DWT this scheme is robust against several attacks. Software model is design by using MATLAB. There is no noticeable difference between the watermarked video frames and the original frames. As a futurework MATLAB simulink model can be interface with FPGA by using Xilinx system generator block.

REFERENCES

1)Snehal V. Patel, Prof. Arvind R. Yadav "Invisible Digital Video Watermarking Using 4-level DWT" National Conference on Recent Trends in Engineering & Technology, 14 May 2011

2) SanjanaSinha, PrajnatBardhan, SwarnaliPramanick, AnkulJagatramka, Dipak K. Kole, ArunaChakraborty.."Digital Video Watermarking using Discrete Wavelet Transform and Principal Component Analysis"International Journal of Wisdom Based Computing, Vol. 1 (2), August 2011

3) HananeMirza, Hien Thai, and ZenshoNakao,"Digital Video Watermarking Based on RGB Color Channels and Principal Component Analysis", KES 2008, Part II, LNAI 5178, pp. 125–132, 2008.

4) Nisreen I. Yassin, Nan cy M. Salem, and Mohamed I. El Adawy "Block Based Video Watermarking Scheme Using Wavelet Transform and Principle Component Analysis"IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 1, No 3, January 2012