Robust Digital Image Watermarking using DCT based Pyramid Transform via image compression

Group 6:45-7:00
Tianlin Yang 40010303
Michael Toledo 40051125

Department of Computer Science
and Software Engineering
Concordia University

Dec,3th, 2019

Content

1. Introduction
2. DCT based Pyramid Transform
3. Validation
4. Improvement
5. Conclusion

1. Introduction - Problem

Multimedia copyright is an critical problem for all of publisher.

Idea: **Can an image contain an 'ID' to protect itself?**

I'm Lena™®©

1. Introduction - Watermark

Idea: **Can an image contain an 'ID' to protect itself?**

Yes, **add watermark!** Simple and easy!

I'm Lena™®©

Watermark

1. Introduction - Watermark

More ugly on the Lena face ?

?

Invisible

Where is the watermark???

1. Introduction - Watermark

Solution: invisible watermark! A little bit tricky! (Transparency <10%)

How?

Extract Data

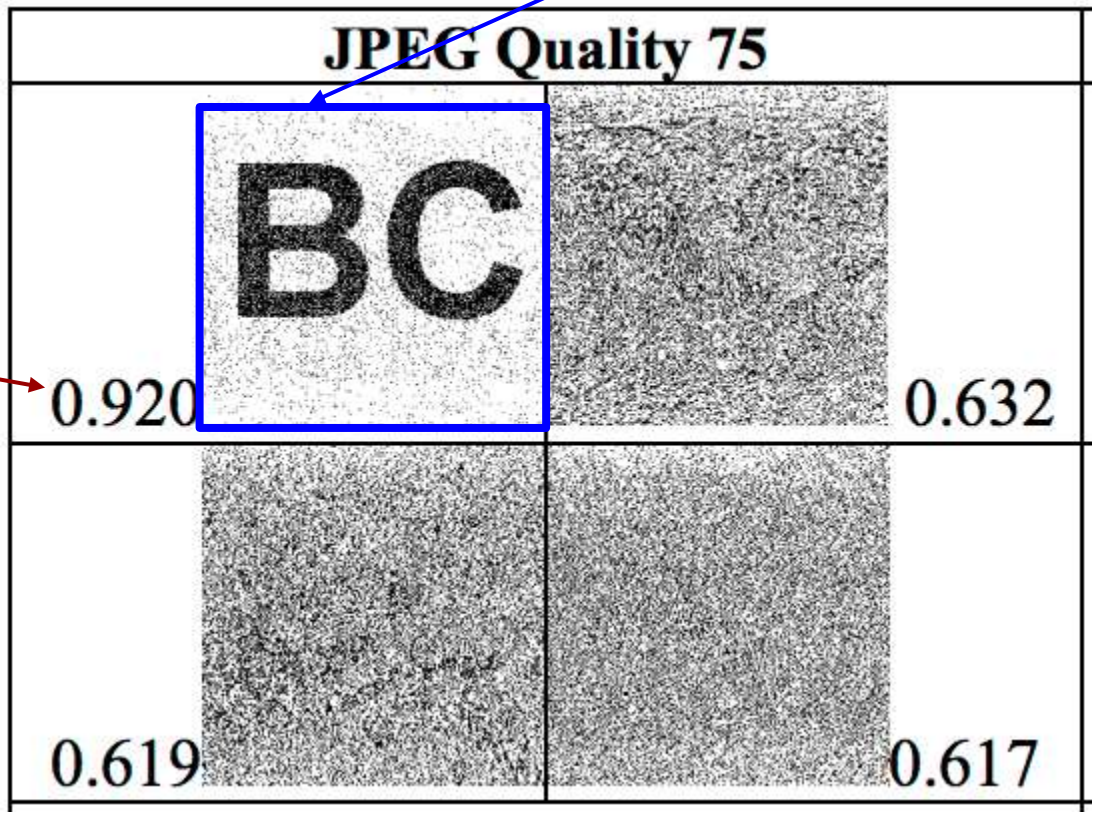
DCT Based Pyramid Transform

A Robust Digital Image Watermarking Algorithm

2. DCT based Pyramid Transform - DCT

First Idea:(*Peining Taoa and Ahmet M. Eskicioglub* 2004 cited by 223)

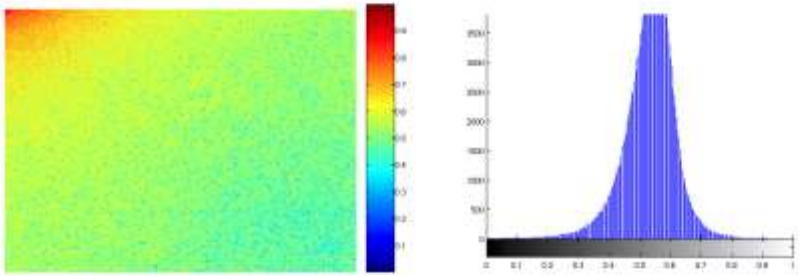
"Watermark data inserted into **low frequencies** is **more robust.**"



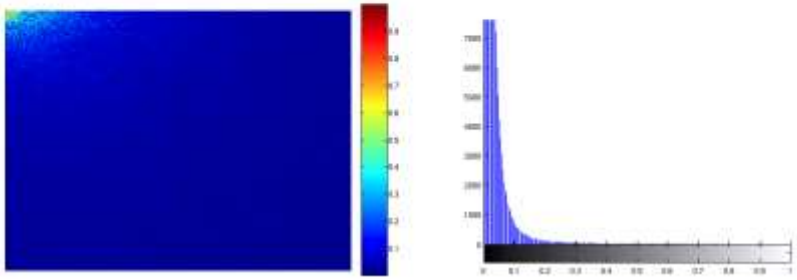
Extracted watermark quality, higher is better.

2. DCT based Pyramid Transform - DCT

DFT



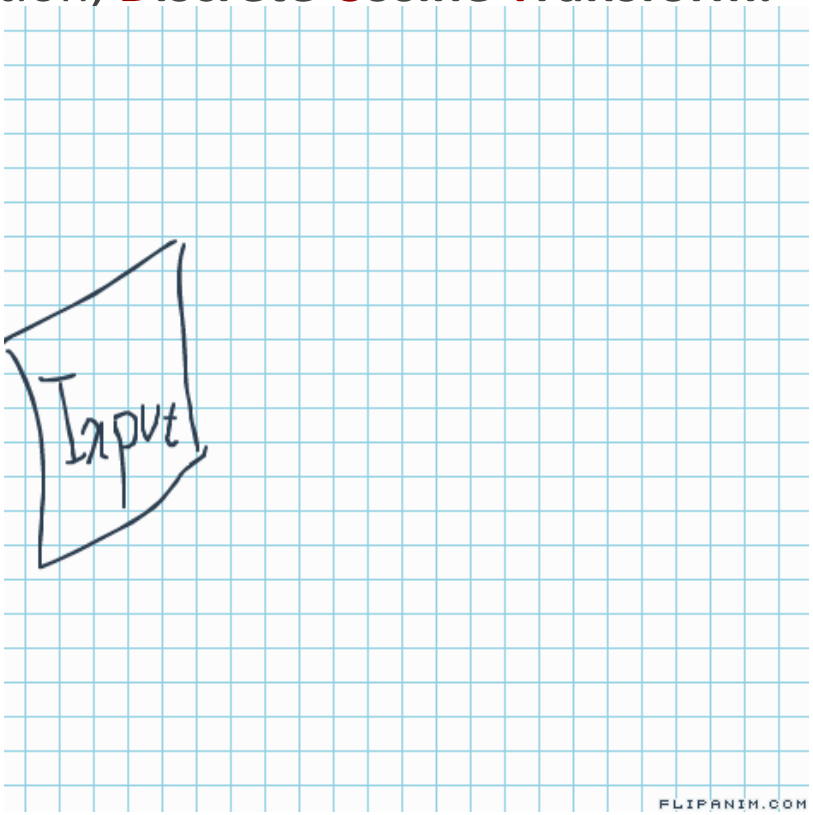
DCT



2. DCT based Pyramid Transform - DCT

How to make the invisible watermark?

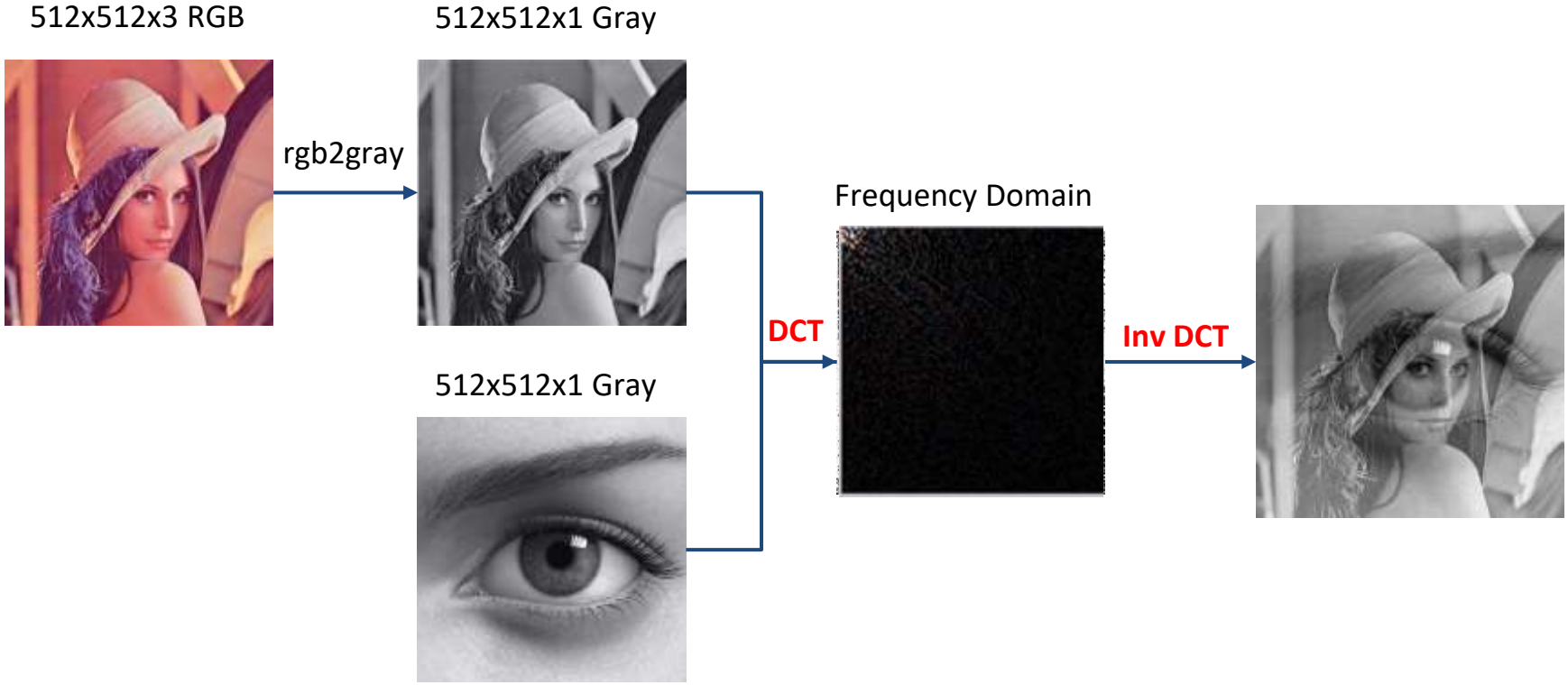
The common solution, **Discrete Cosine Transform.**



2. DCT based Pyramid Transform - DCT

How to make the invisible watermark?

The common solution, **Discrete Cosine Transform.**



2. DCT based Pyramid Transform - DCT

DCT-only can do a good job with transparency < 10% watermarking,

DCT-only watermark result: alpha=0.2, q=80

2. DCT based Pyramid Transform - Extract

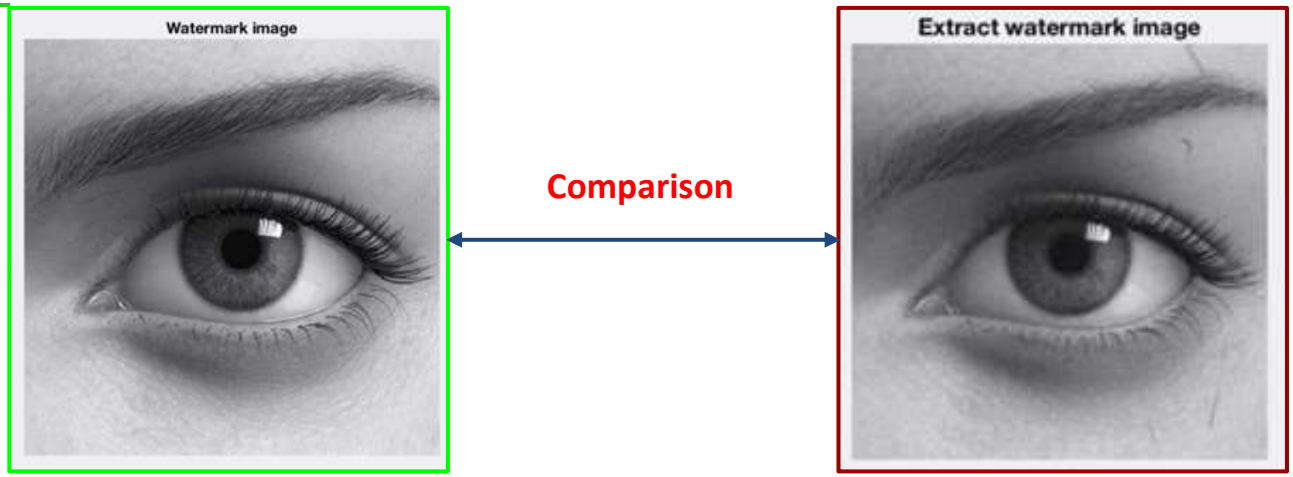
DCT Watermark extract:

Extracted watermark = Watermarked image - original image

2. DCT based Pyramid Transform - PSNR

PSNR: (Peak signal to noise Ratio) is the answer, **the higher, the better!**

DCT-only watermark result: alpha=0.2, q=80




```
% Root mean square error
[m,n] = size(imtarget);%get the size of input image m,n
RMSE = sqrt(sum((imtarget(:)-imwatermark(:)).^2)/(m*n));
fprintf('\n The RMSE is %0.4f\n', RMSE);
% Peak signal to noise Ratio
L = 255;% uint8 should be 255
PSNR = 10*log10(L^2/RMSE);
fprintf('\n The PSNR is %0.4f\n', PSNR);
```

```
>> DCTonly
Transparency of watermarking alpha = 0.2
Quality Factor q = 80

 The RMSE is 122.5637

The PSNR is 27.2472
```

2. DCT based Pyramid Transform - Code

DCT-Only Code: Ok, let's implement it **line-by-line**.

```
Editor - /Users/lin/Desktop/IMproject 2/TianLin Yang/DCTonly.m
DCTonly.m
 6 -   clear all;
 7     %Set transparency ratio of the watermark
 8 -   alpha=input('Transparency of watermarking
 9     %Ask user input the Quality Factor
10 -   q=input('Quality Factor q = ');
11
12    %Import images
13 -   imtarget = double(imread('LenaGray.bmp'));
14 -   imwatermark = double(imread('WaterMarkSree
15    %Plot our inputs
16 -   figure(1);
17 -   subplot(1,4,1),
18 -   imshow(imtarget,[]);
19 -   title 'Input image';
20 -   subplot(1,4,2),
21 -   imshow(imwatermark,[]);
22 -   title 'Watermark image';
23
24    %Apply Basic DCT2 process for inputs
25 -   imTrans = dct2(imtarget);%Trans our input
26 -   wtTrans = dct2(imwatermark);%Trans waterma
27 -   CombineIM=imTrans+alpha*wtTrans;%Combine w
28    %plot the combined output image
29 -   OutputI=idct2(CombineIM);
30 -   subplot(1,4,3),
29 -   OutputI=idct2(CombineIM);
30 -   subplot(1,4,3),
31 -   imshow(OutputI,[]);
32 -   title 'DCT-Watermarked output image';
33
34    %Use quality factor generate output image.
35 -   imwrite(uint8(OutputI),'WatermarkedOutput.jpg','jpg','quality',
36 -   compr=imread('WatermarkedOutput.jpg');%read output.jpg as our c
37    %Extract the watermark
38 -   extractWatermark=CombineIM-imTrans;
39 -   EWTimage = idct2(extractWatermark);
40 -   subplot(1,4,4),
41 -   imshow(EWTimage,[]);
42 -   title 'Extract watermark image';
43    %Get the reference score to make comparations
44 -   imtarget=imwatermark;
45 -   imwatermark=EWTimage;
46    % Root mean square error
47 -   [m,n] = size(imtarget);%get the size of input image m,n
48 -   RMSE = sqrt(sum((imtarget(:)-imwatermark(:)).^2)/(m*n));
49 -   fprintf('\n The RMSE is %0.4f\n', RMSE);
50    % Peak signal to noise Ratio
51 -   L = 255;% uint8 should be 255
52 -   PSNR = 10*log10(L^2/RMSE);
53 -   fprintf('\n The PSNR is %0.4f\n', PSNR);
```

```
Command Window
  Transparency of watermarking alpha = 0.2
  Quality Factor q = 80

   The RMSE is 122.5637

   The PSNR is 27.2472
fx >>
```

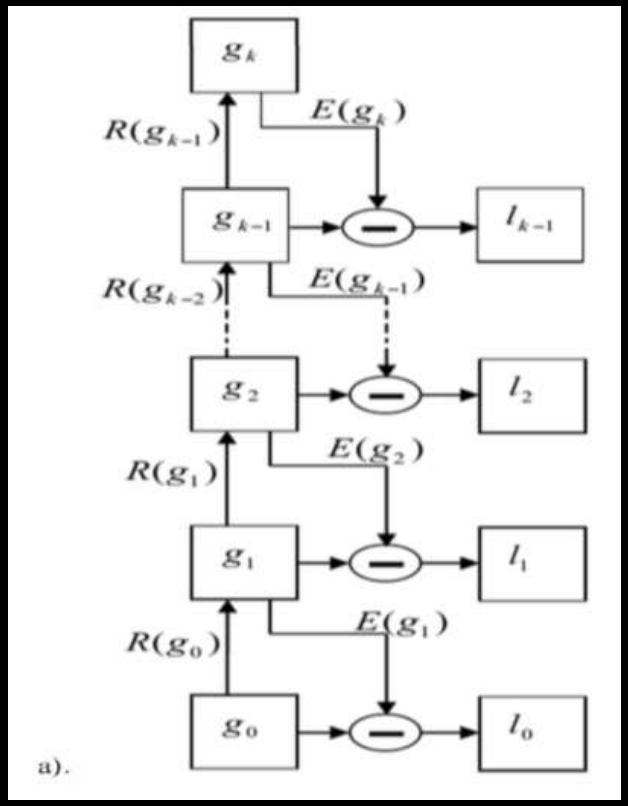
2. DCT based Pyramid Transform - Pyramid

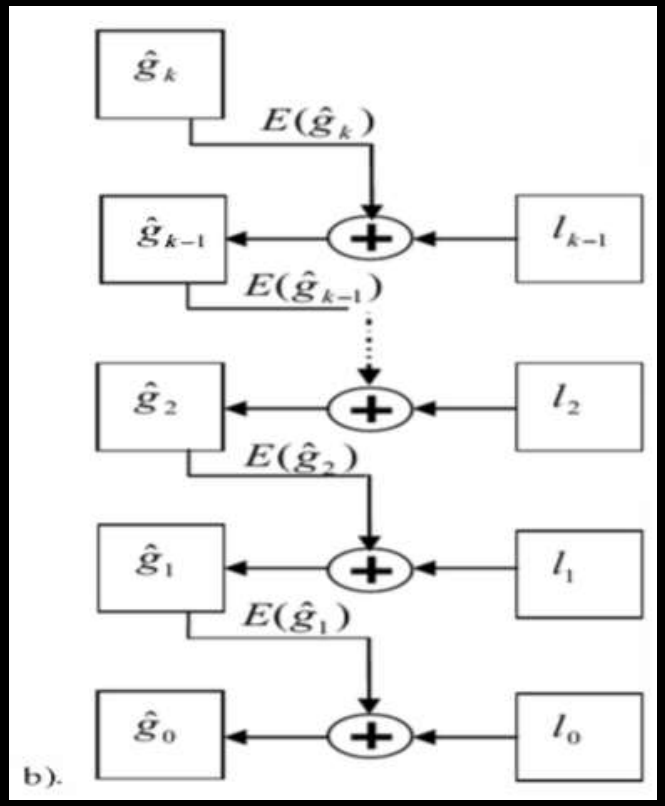
DCT-only is indeed great for watermarking.

But is there anything better?

2. DCT based Pyramid Transform - Pyramid

Pyramid structure is a good try:

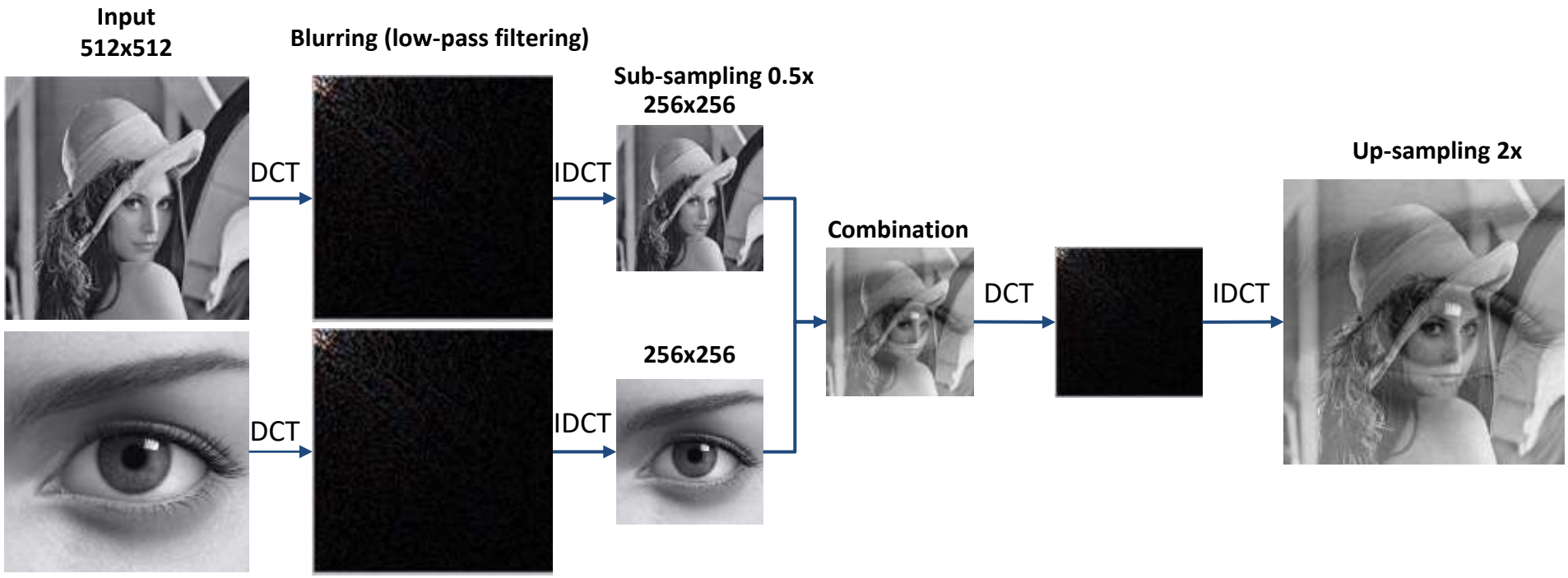




What is this???

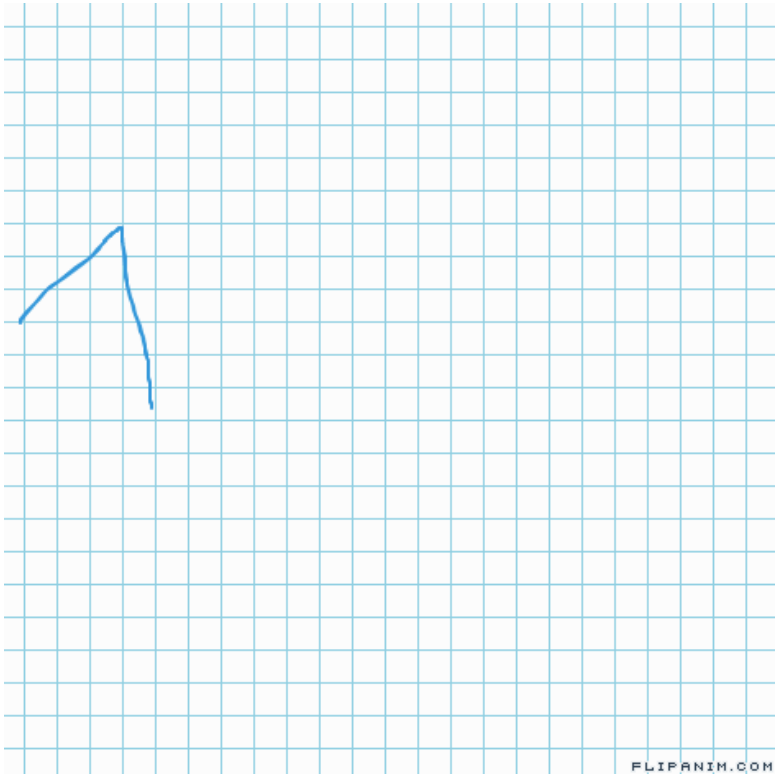
2. DCT based Pyramid Transform - Pyramid

Laplacian Pyramid: Each level of pyramid is recursively constructed from lower level:



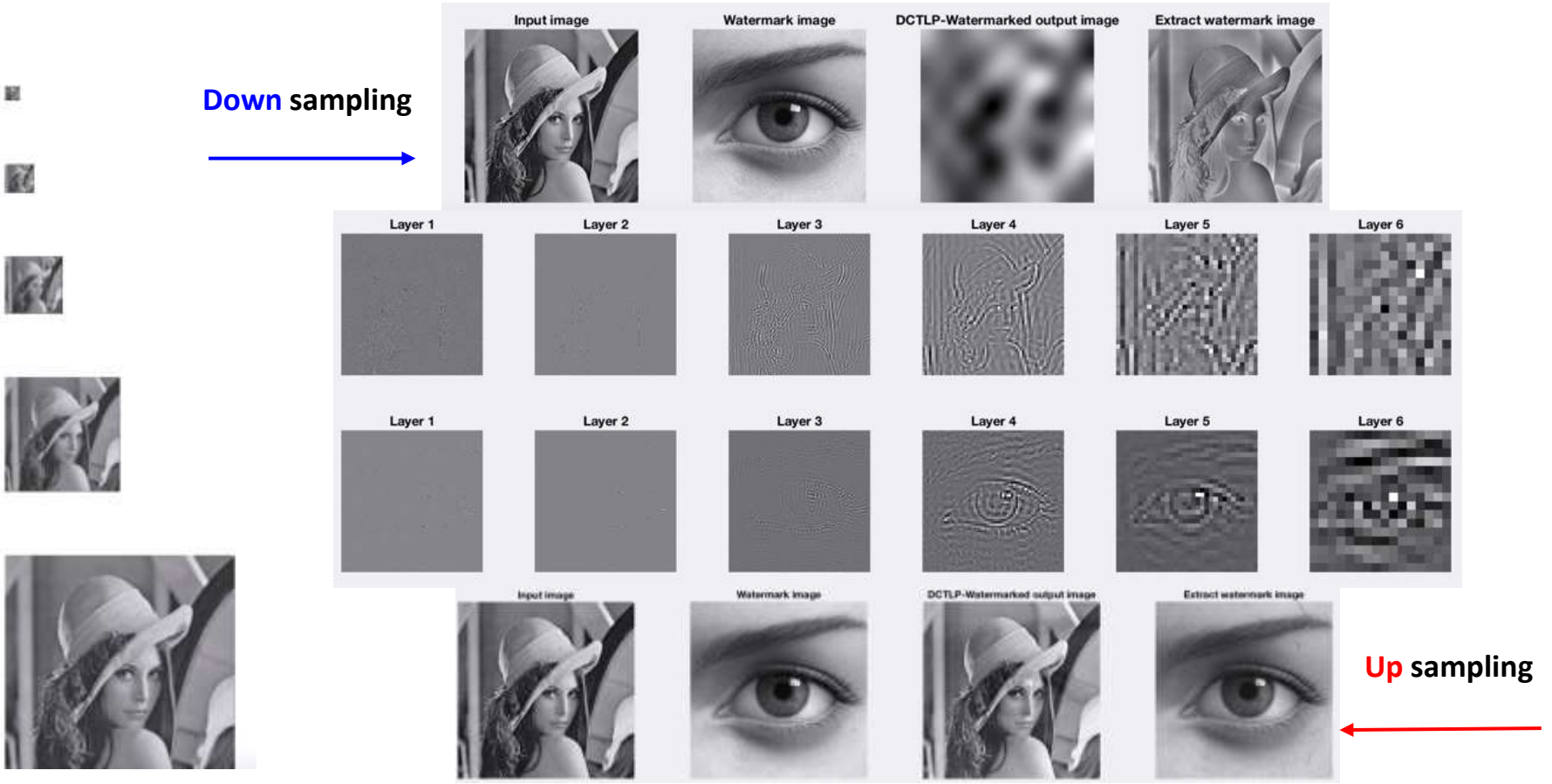
2. DCT based Pyramid Transform - Pyramid

Laplacian Pyramid: Each level of pyramid is recursively constructed from **lower level**: **Animation time!**



2. DCT based Pyramid Transform - Pyramid

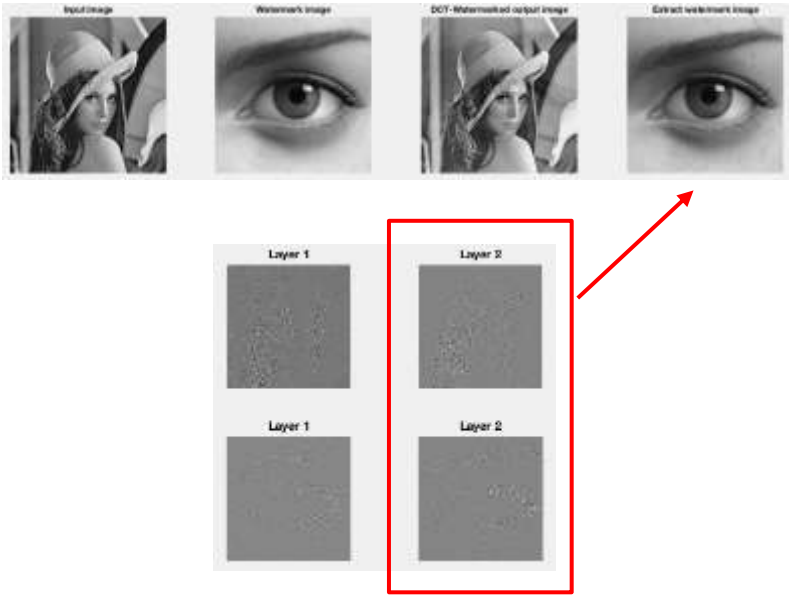
Laplacian Pyramid: is via image **Compression.**



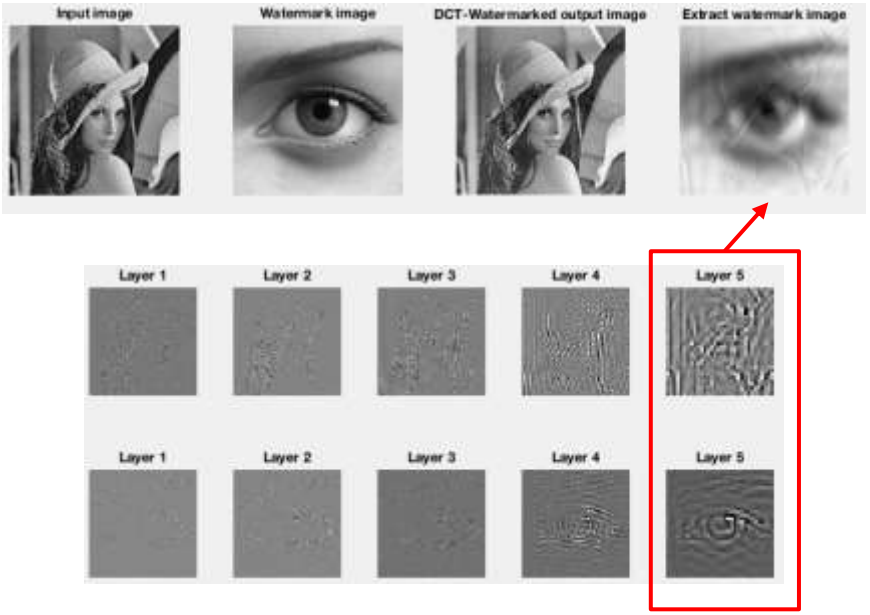
2. DCT based Pyramid Transform - Pyramid

Laplacian Pyramid: Each level of pyramid is recursively constructed from **lower level**: compression than combine, with high PSNR output.

2-level pyramid

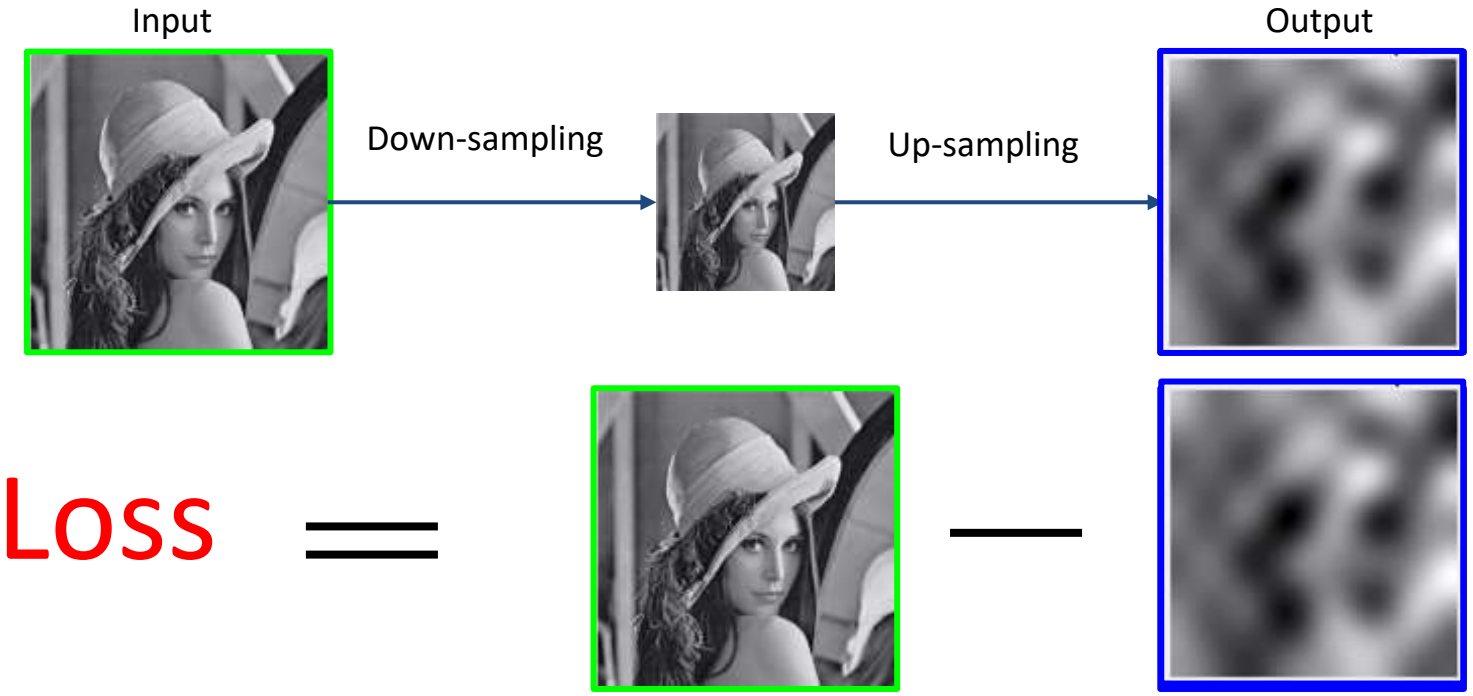


5-level pyramid



2. DCT based Pyramid Transform - Pyramid

loss : Recovery image to original 512x512 by add the loss.

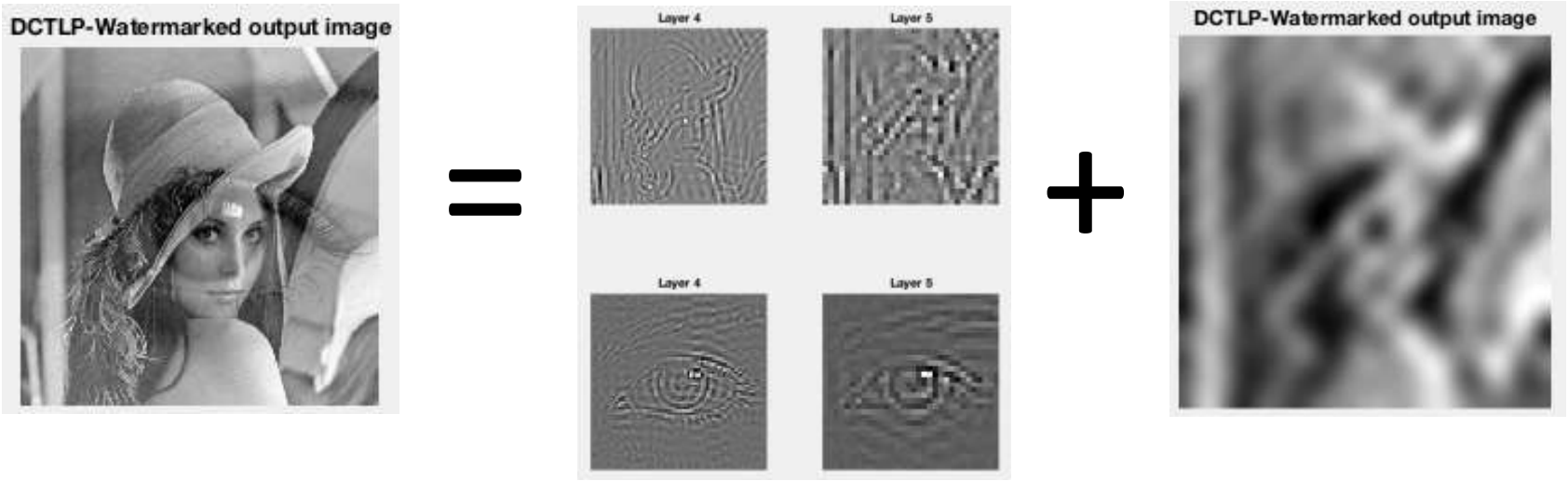


2. DCT based Pyramid Transform - Pyramid

Recovering: By add the loss back to the combined output.

Loss

Lowest-level combination



2. DCT based Pyramid Transform - Pyramid

Laplacian Pyramid: Each level of pyramid is recursively constructed from **lower level**: **Code line by line** add **loss** in the output

```
%DCT-PT
function[imf,Idf,Idf2] = DCTPT(InputIM,IMwaterMark,alpha1,k)
%Image down-sampling
for i = 1:k
    IM = reduce2d(InputIM);%down-sampling inputIM with Gaussian pyramid
    Idf{i} = InputIM - expand2d(IM);%Laplacian pyramid loss for input image
    InputIM = IM;% Now update to InputIM

    IM = reduce2d(IMwaterMark);%down-sampling watermark with Gaussian pyramid
    Idf2{i} = IMwaterMark - expand2d(IM);%Laplacian pyramid loss for input image
    IMwaterMark = IM;% Now update to watermark
end
imw=InputIM+alpha1*IMwaterMark;%Combine them
%Image reconstruction
imf = imw;
for i=k:-1:1
    imf = Idf{i}+alpha1*Idf2{i}+ expand2d(imf);% add loss back, also add alpha fc
end

end
```

2. DCT based Pyramid Transform - Compare

DCTPT is indeed the **better** approach: For alpha=0.01 q=80

DCT-only

The PSNR is 23.3192

VS

DCT+Pyramid

The PSNR is 27.5540

3. Validation

So far so good, time to **validate our program** with author's result!

Paper results		
Alpha	q (quality factor)	DCTPT (PSNR)
0.2	80	37.4035
0.1	80	34.7033
0.05	80	31. 6964
0.03	80	29. 5015
0.01	80	24.7303

Our results		
Alpha	q (quality factor)	DCTPT (PSNR)
0.2	80	39.9276
0.1	80	37.2569
0.05	80	34.5291
0.03	80	32.3182
0.01	80	27.5540

3. Validation - Sensitive rate test with alpha

So far so good, time to **validate our program** with author's result!

Our result shows **better PSNR** with same ratio change.

Paper results		
Alpha	q (quality factor)	DCTPT (PSNR)
0.2	80	37.4035
0.1	80	34.7033
0.05	80	31. 6964
0.03	80	29. 5015
0.01	80	24.7303

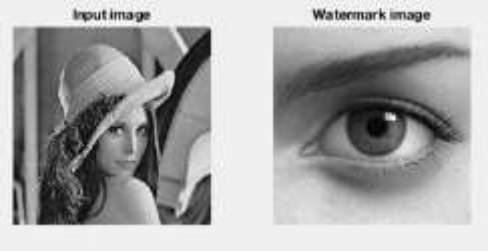
+3 for input difference, than it's Same

Our results		
Alpha	q (quality factor)	DCTPT (PSNR)
0.2	80	39.9276
0.1	80	37.2569
0.05	80	34.5291
0.03	80	32.3182
0.01	80	27.5540

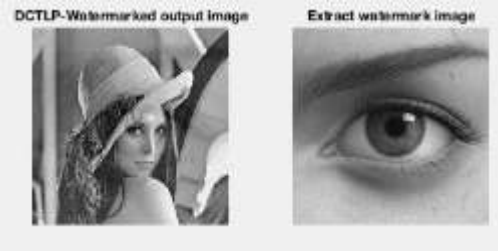
3. Validation - Sensitive rate test with alpha

Extract watermark quality: less alpha, less quality(PSNR)

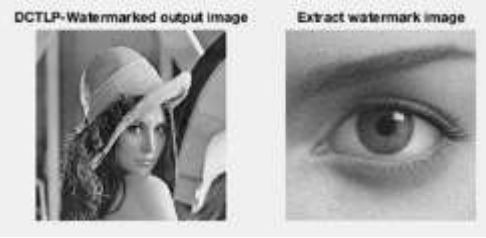
Original



alpha=0.2 q=80



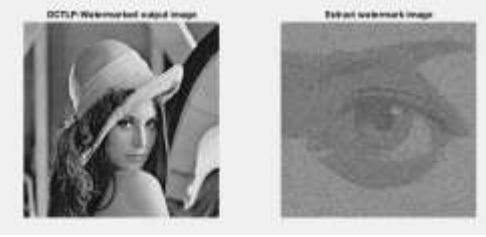
alpha=0.1 q=80



alpha=0.05 q=80

alpha=0.03 q=80

alpha=0.01 q=80



3. Validation - Sensitive rate test with JPEG q

So far so good, time to **validate our program** with author's result!

Our result shows **better PSNR** with same ratio change.

Paper results		
Alpha	q (quality factor)	DCTPT (PSNR)
0.05	30	27.4307
0.05	50	28.9136
0.05	70	30.4795
0.05	80	31. 6964
0.05	90	34.0585
005	100	38.1264

+3 for input difference, than it's Same

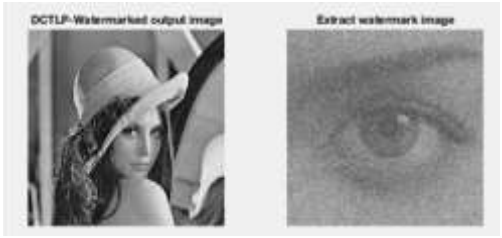
Our results		
Alpha	q (quality factor)	DCTPT (PSNR)
0.05	30	30.3926
0.05	50	31.8473
0.05	70	33.3560
0.05	80	34.5291
0.05	90	36.6017
005	100	39.5801

3. Validation - Sensitive rate test with JPEG q

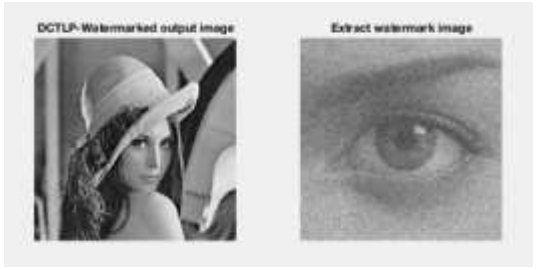
Extract watermark quality: more qualityFactor(q), more quality(PSNR)

Original

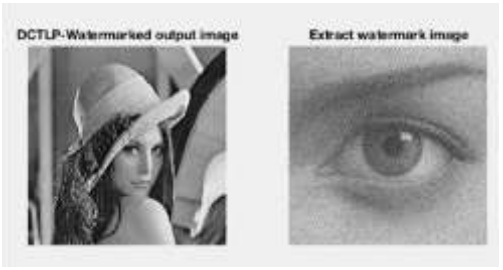
alpha=0.05 q=30



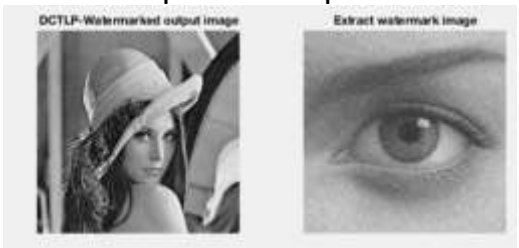
alpha=0.05 q=50



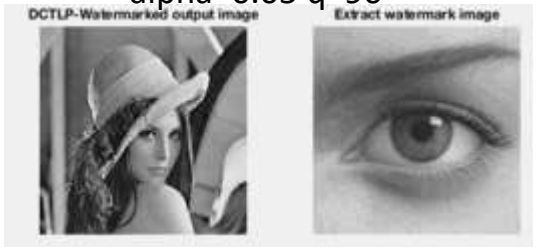
alpha=0.05 q=70



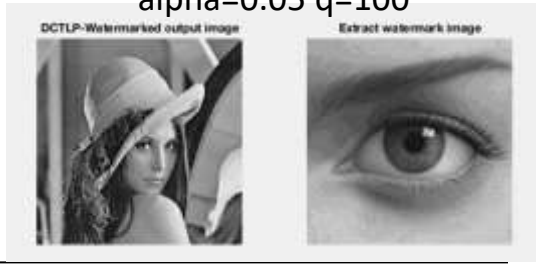
alpha=0.05 q=80



alpha=0.05 q=90



alpha=0.05 q=100



3. Validation - Problems

- Only considered the Gray **image**, not for **RGB** image.
- The paper never simulate some **Attacks** to validate the robustness of this algorithm.

$$PSNR = 20\log_{10}\left(\frac{L^2}{\frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}\left(I_r(x,y) - I_f(x,y)\right)^2}\right)$$

Original image is lena.jpg as shown fig. 3.

Fig. 3. Original image

Embedding watermark image gray4.jpg as shown fig. 4.

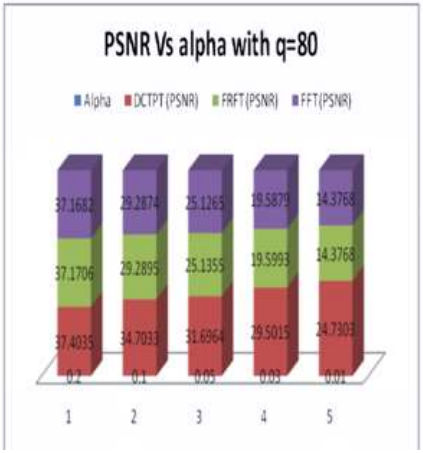


Fig. 4. Watermark image

TABLE I

VARY ALPHA AND CONSTANT Q

Alpha	q(quality factor)	DCTPT (PSNR)	FRFT (PSNR)	FFT (PSNR)
0.2	80	37.4035	37.1706	37.1682
0.1	80	34.7033	29.2895	29.2874
0.05	80	31.6964	25.1355	25.1265
0.03	80	29.5015	19.5993	19.5879
0.01	80	24.7303	14.3768	14.3768

Fig. 5. chart for PSNR v alpha and constant q

TABLE 2

CONSTANT ALPHA AND VARY Q

Alpha	q(quality factor)	DCTPT (PSNR)	FRFT PSNR	FFT PSNR
0.05	30	27.4307	24.0136	24.0078
0.05	50	28.9136	24.3145	24.3127
0.05	70	30.4795	25.1172	25.1094
0.05	80	31.6964	25.1355	25.1265
0.05	90	34.0585	25.1732	25.1674
0.05	100	38.1264	25.2007	25.1954

4. Improvement - RGB

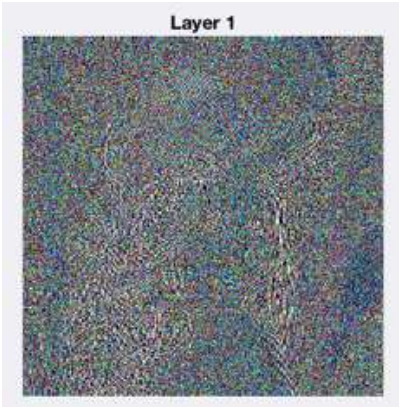
- Ok, now first finish the **RGB** part: alpha=0.02 q=80

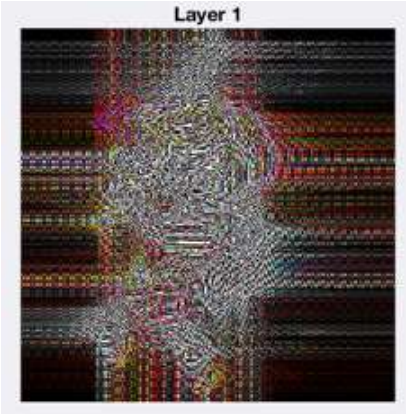
```
%RGB DCTPT
function[imf,Idf,Idf2] = DCTPT(InputIM,IMwaterMark,alpha1,k)
InputIM = double(InputIM);
IMwaterMark = double(IMwaterMark);
for i = 1:k
    IM = reduce2d(InputIM);%down-sampling inputIM with Gauss
    Idf{i} = InputIM - expand2d(IM);%Laplacian pyramid loss
    InputIM = IM;% Now update to InputIM

    IM = reduce2d(IMwaterMark);%down-sampling watermark with
    Idf2{i} = IMwaterMark - expand2d(IM);%Laplacian pyramid
    IMwaterMark = IM;% Now update to watermark
end
    %RGB combination
    conb(:,:,1)=InputIM(:,:,1)+alpha1*IMwaterMark(:,:,1);
    conb(:,:,2)=InputIM(:,:,2)+alpha1*IMwaterMark(:,:,2);
    conb(:,:,3)=InputIM(:,:,3)+alpha1*IMwaterMark(:,:,3);
```

4. Improvement - RGB

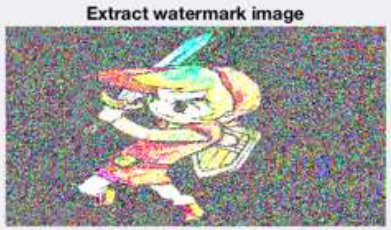
- Ok, now first finish the **RGB** part: alpha=0.02 q=80

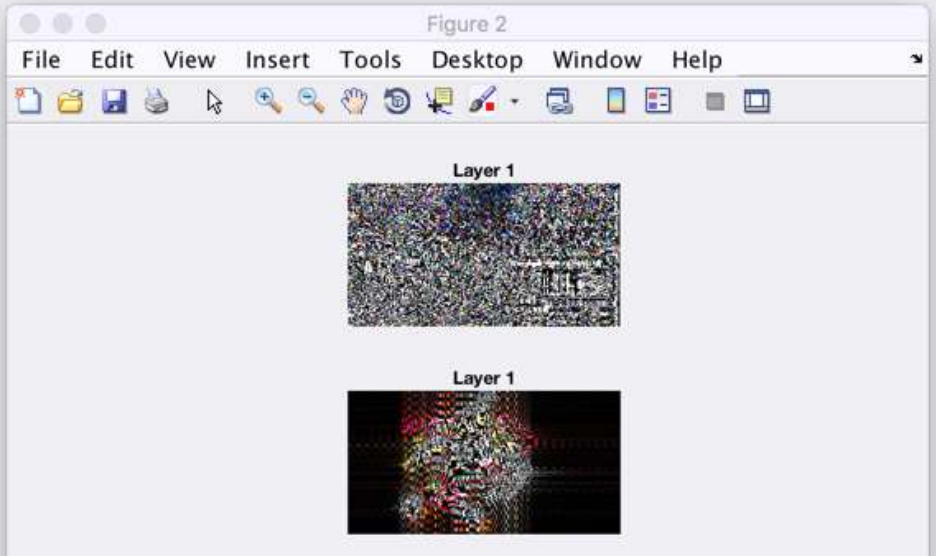




4. Improvement - RGB

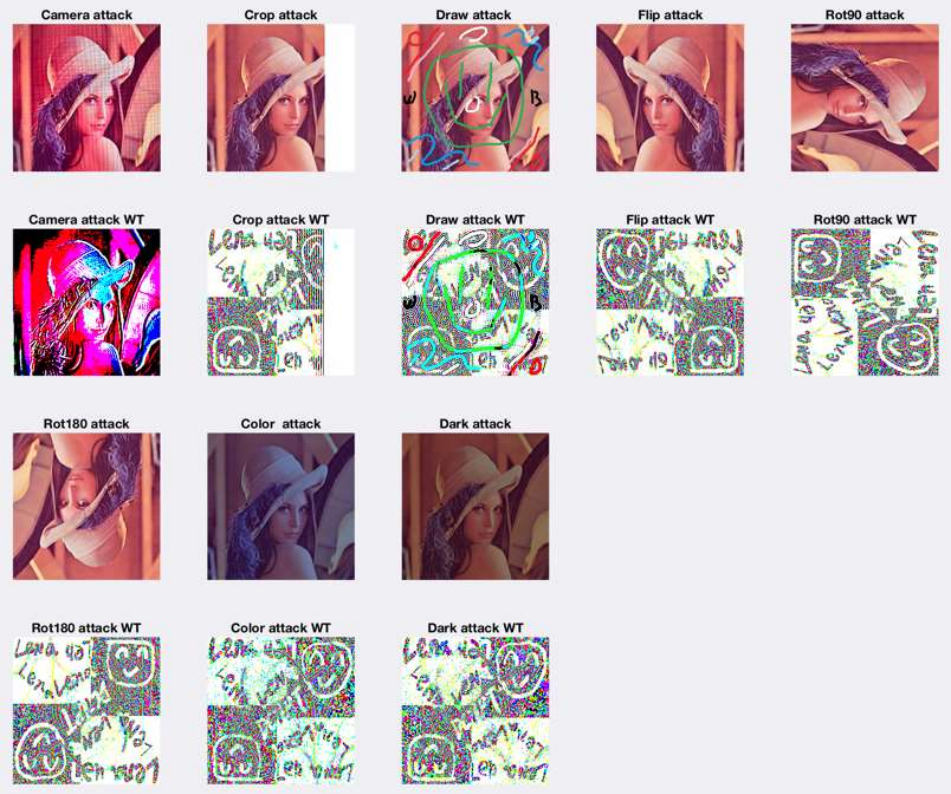
- Ok, now first finish the **RGB** part: alpha=0.02 q=80



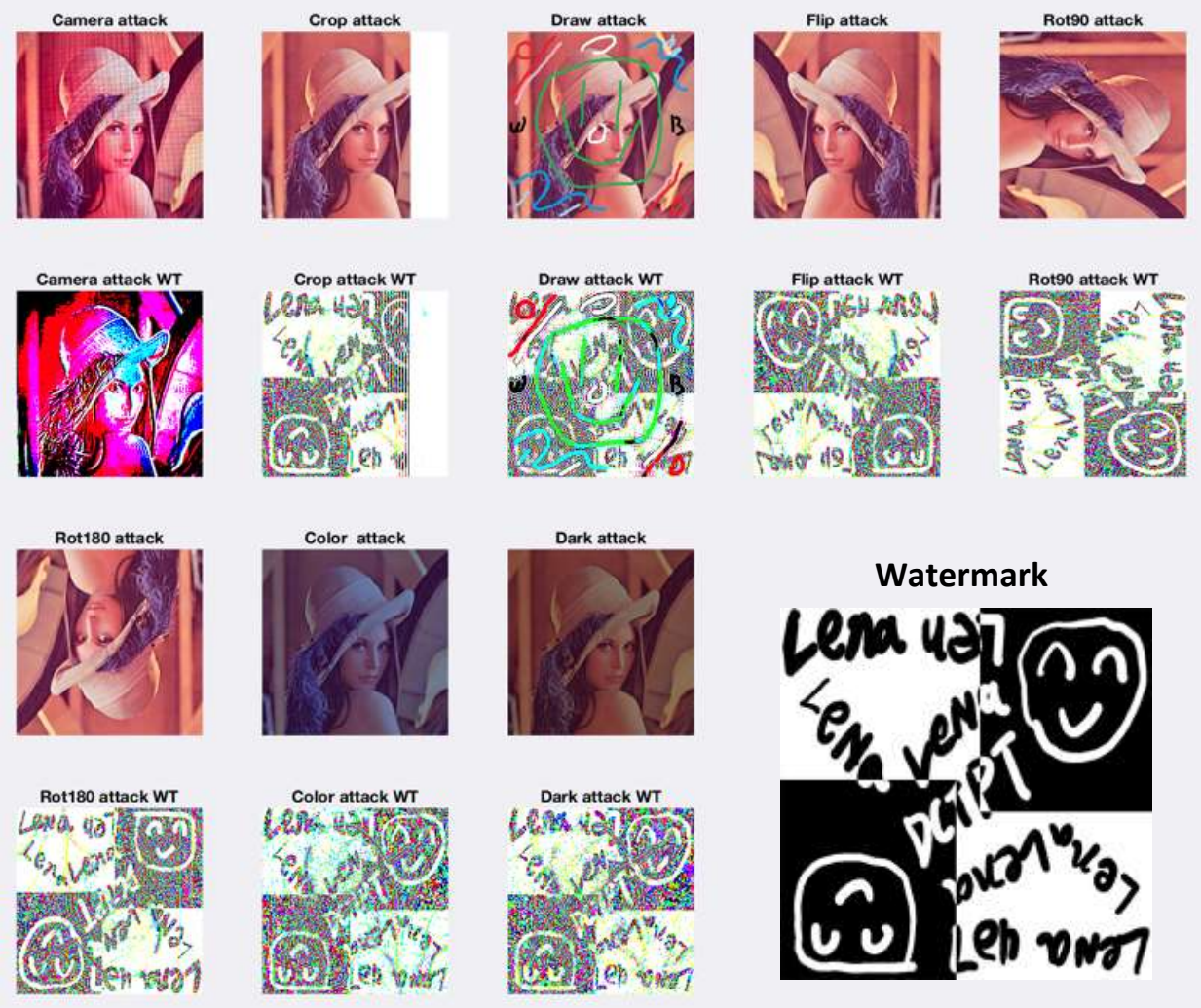


4. Improvement - Attack simulation

- The model shows great robustness on rotation, compression, flip, crop, etc.



4. Improvement - Attack simulation



4. Improvement - Attack simulation

- The **drawing** is replace some of pixels, so get **bad** **result**,
- **Camera** attack is the **Hardest one** to handle.

Camera attack

Camera attack WT

Any solution for them???

Draw attack

Draw attack WT

4. Improvement - Attack simulation

- The **Smart-Toy** with watermark on it at **2002**:
- "Digital watermarking is a complex technology necessarily involving many conflicting **requirements and tradeoffs**."
- R.K. Sharma et al,"Practical Challenges for Digital Watermarking Applications", **2002**

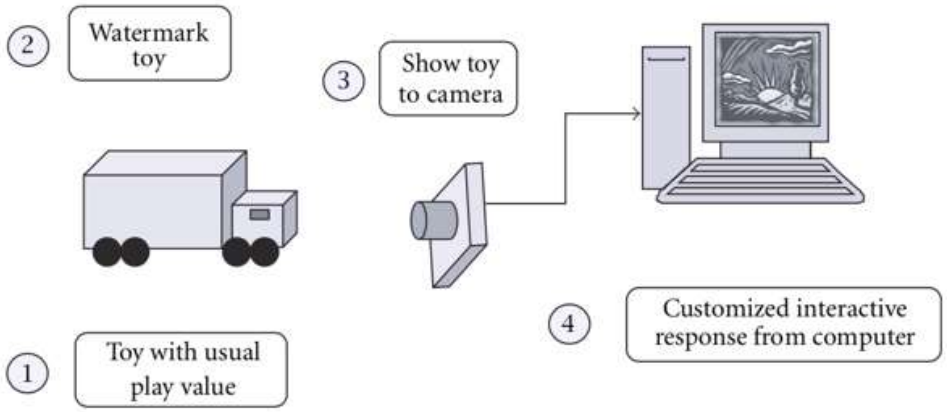


FIGURE 1: Illustration of the Smart Toy concept.

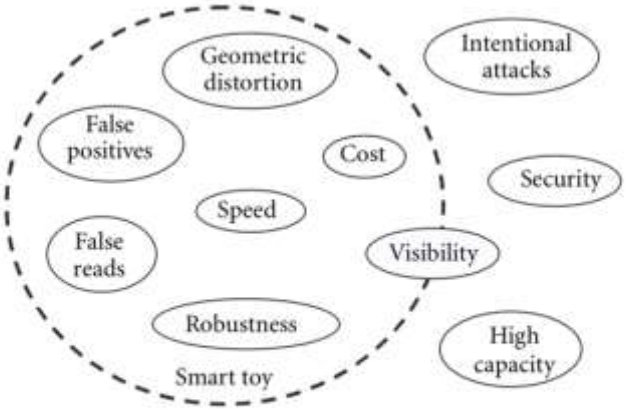


FIGURE 2: How Smart Toy requirements fit in the general space of watermarking requirements.

4. Improvement - Attack simulation

- The **trade-off triangle** for watermarking at **2007**:
- "Lot of problems, however, wait to be solved."
- A. Pramila et al,"Camera based watermark extraction – problems and examples", **2007**

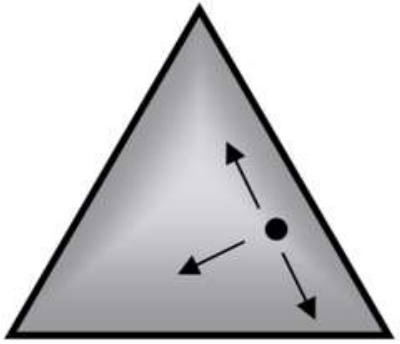


Figure 1. Triangle of trade offs while choosing watermark properties.

Figure 4. Searching the corner points.

4. Improvement - Solution for camera attack

*K Thongkor,"Digital watermarking for camera-captured images based on just noticeable distortion and Wiener filtering",**2018***

This is indeed an good solution for camera attack, and they get great result, complex processes:

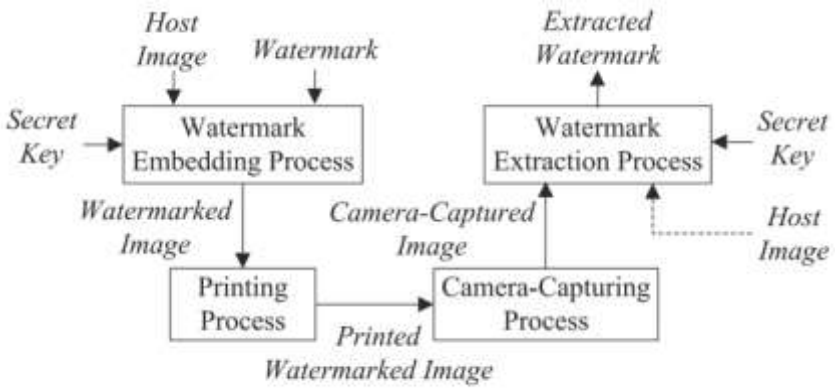


Fig. 2. Overview of image watermarking for camera-captured images.

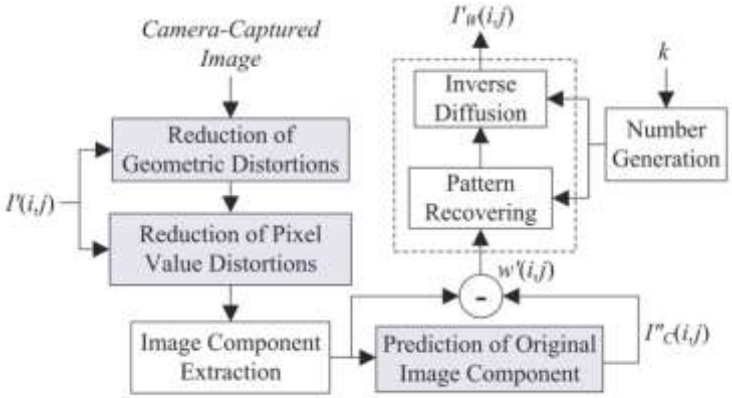
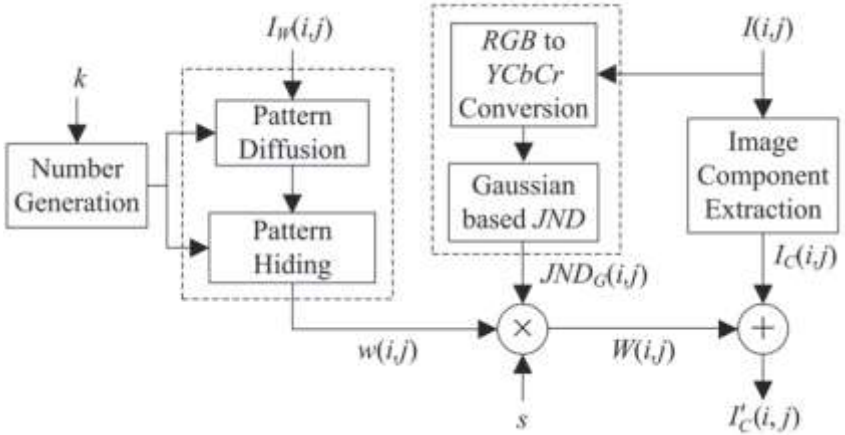


Fig. 5. Block diagram of the proposed watermark extraction process.



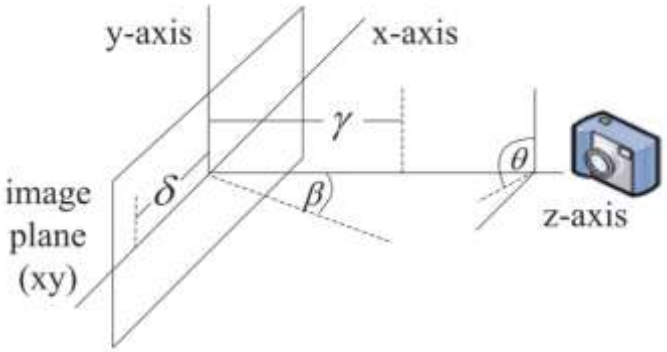
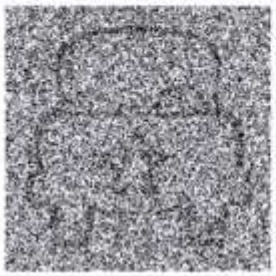
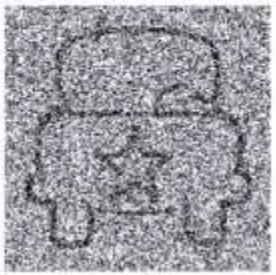


Fig. 11. The viewpoint of camera with reference axes.

4. Improvement - Solution for camera attack

"Digital watermarking for camera-captured images based on just noticeable distortion and Wiener filtering"

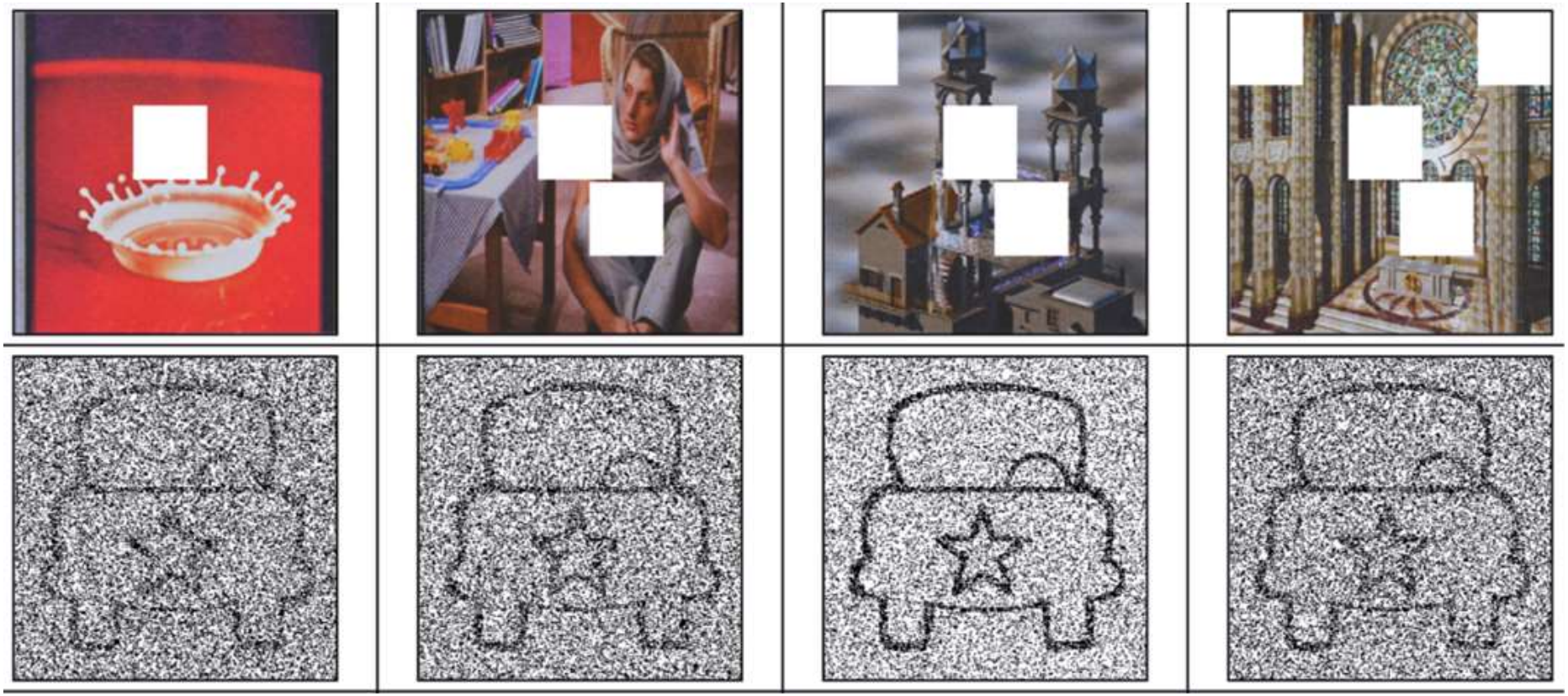
This is indeed an good solution for camera attack, and they get great result:

The watermarked image (256×256)				
The captured image (scaled and cropped from 3456×2304)				
The extracted watermark				

4. Improvement - Solution for camera attack

"Digital watermarking for camera-captured images based on just noticeable distortion and Wiener filtering"

This is indeed an good solution for camera attack, and they get great result:



4. Improvement - Solution for camera attack

Key idea to protect watermark from **camera** of this paper:

1. To get large amount watermark, **all** image pixels is embedded to carry watermark bit. (watermark should be **binary image 0s 1s**)
2. Strength of watermark is **just noticeable**.
3. By reducing distortion, predict the original image from photo.

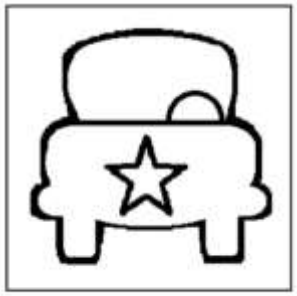


Fig. 9. The original watermark image.

```
n0 = 0, n1 = 0        % n0 and n1 are number of bit 0s and 1s
for each pixel coordinate, (Pi,Pj), in the permutation sequence
  for m = −1 to 1 do
    for n = −1 to 1 do
      if n1 ≥ 5
        RX(Pi + m,Pj + n) = 0
      else if n0 ≥ 5
        RX(Pi + m,Pj + n) = 1
      else
        RX(Pi + m,Pj + n) = genRandomInt[0,1]
      end if
    end for
  end for
end for
```

4. Improvement: Encode watermark

- Well, all computer with Matlab now can get my watermark using DCTPT model.
- What if i want to **hide my secret?** only who get the **'key'** can access.

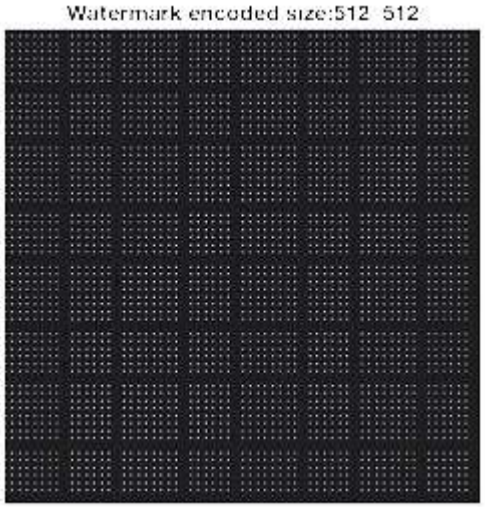
Yes, i get it!

4. Improvement: Encode watermark

- By apply Encode-Decode techniques, the RGB watermark image could be transfer to **binary information** and encode to 8x8 blocks
- By insert to **random position**, the encoded watermark is secured.

RGB image

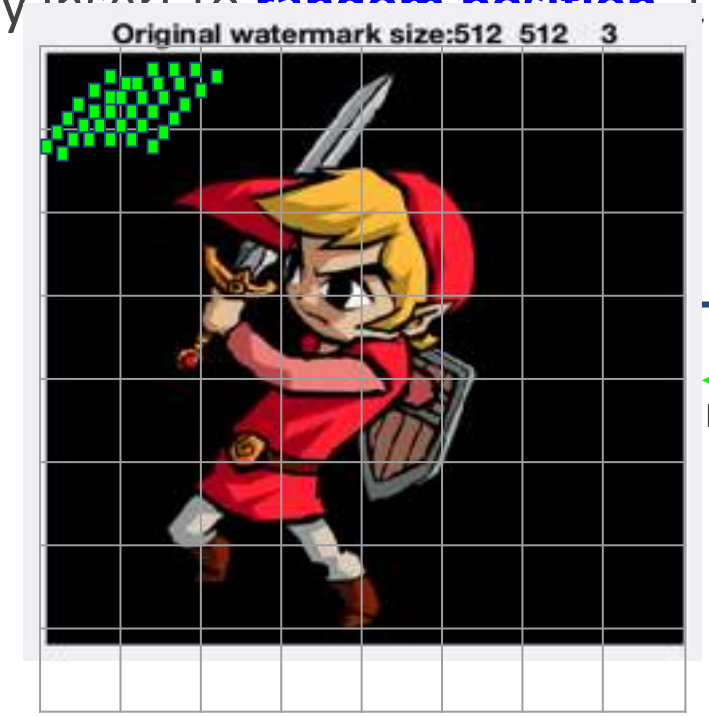
Binary BW image



Encode image

4. Improvement: Encode watermark

- By apply Encode-Decode techniques, the RGB watermark image could be transfer to **binary information** and encode to **8x8 blocks**
- By insert to **random position**, the encoded watermark is secured.



Key for black = [1,2,3,4,5,6,7,8]
Key for white = [2,2,3,3,4,4,8,5]

64x64

8x Down-sampling to 64x64

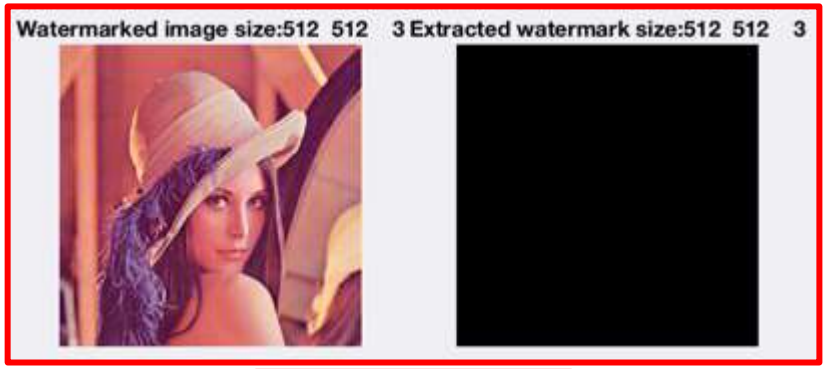
Put **encoded** 512x512 pixels back(**key**)

1 **green** pix get 1 **Key**(8 length)
64 green pixels = 64 x 8 = **512 Key** pixels

4. Improvement: Encode watermark

- When extract the watermark, only **right key** can get information.

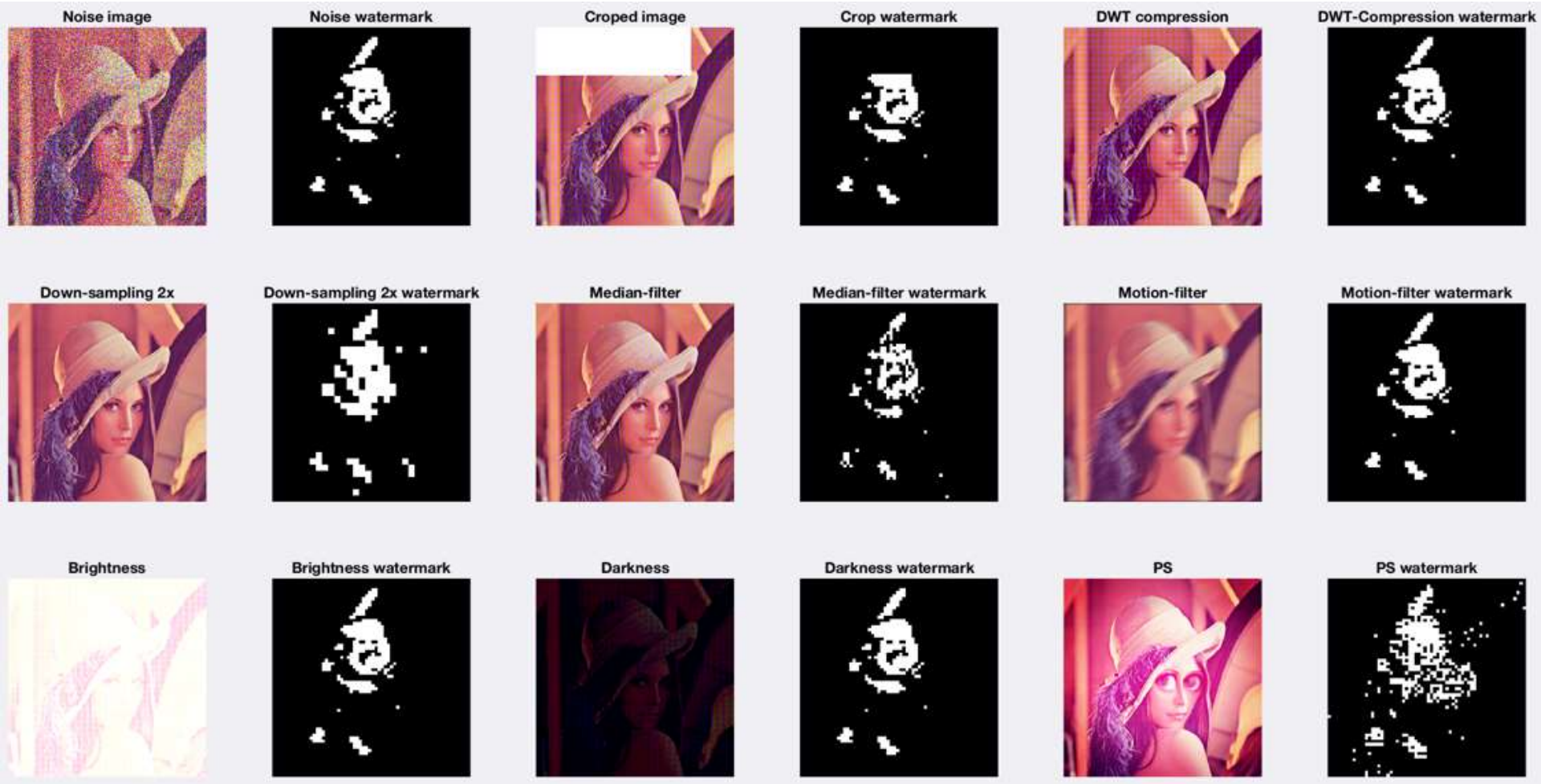
```
%Right Key
k1 = [-0.1255    1.1678   -1.6970   -1.1682   -0.1460    2.4896    0.3280    0.1848];
k2 = [2.1891    0.2804    1.3281    0.8329   -1.7363   -0.0759    2.0774    1.1366];
%If i use another key?
k1 = [2.1891    0.2804    1.3281    0.8329   -1.7363   -0.0759    2.0774    1.1366];
k2 = [2.1891    0.2804    1.3281    0.8329   -1.7363   -0.0759    2.0774    1.1366];
```



4. Improvement: Encode watermark

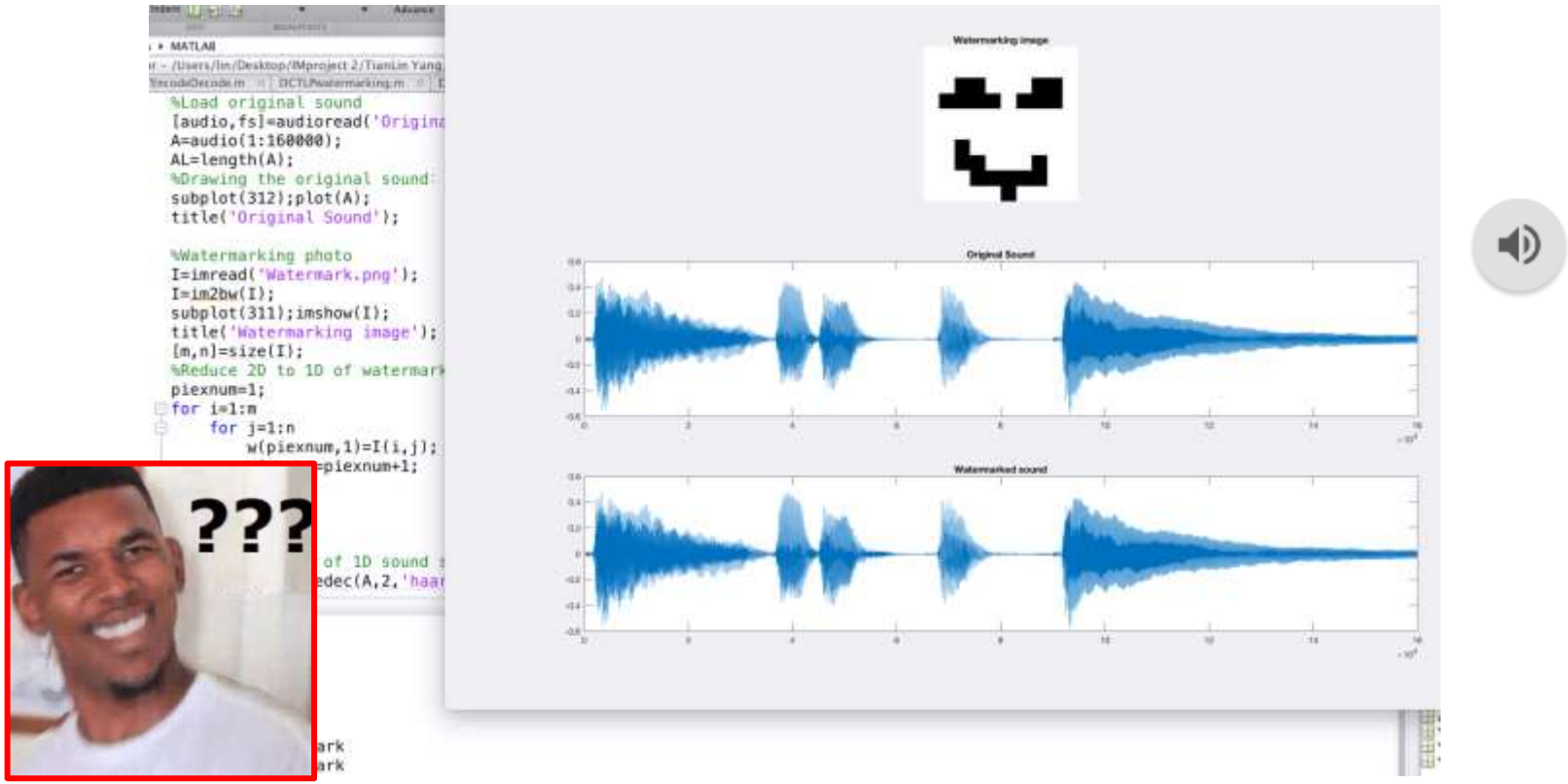
- By apply Encode-Decode techniques, the RGB watermark image could be transfer to **binary information** and encode to 8x8 blocks
- By insert to **random position**, the encoded watermark is secured.

4. Improvement: More Attack tests



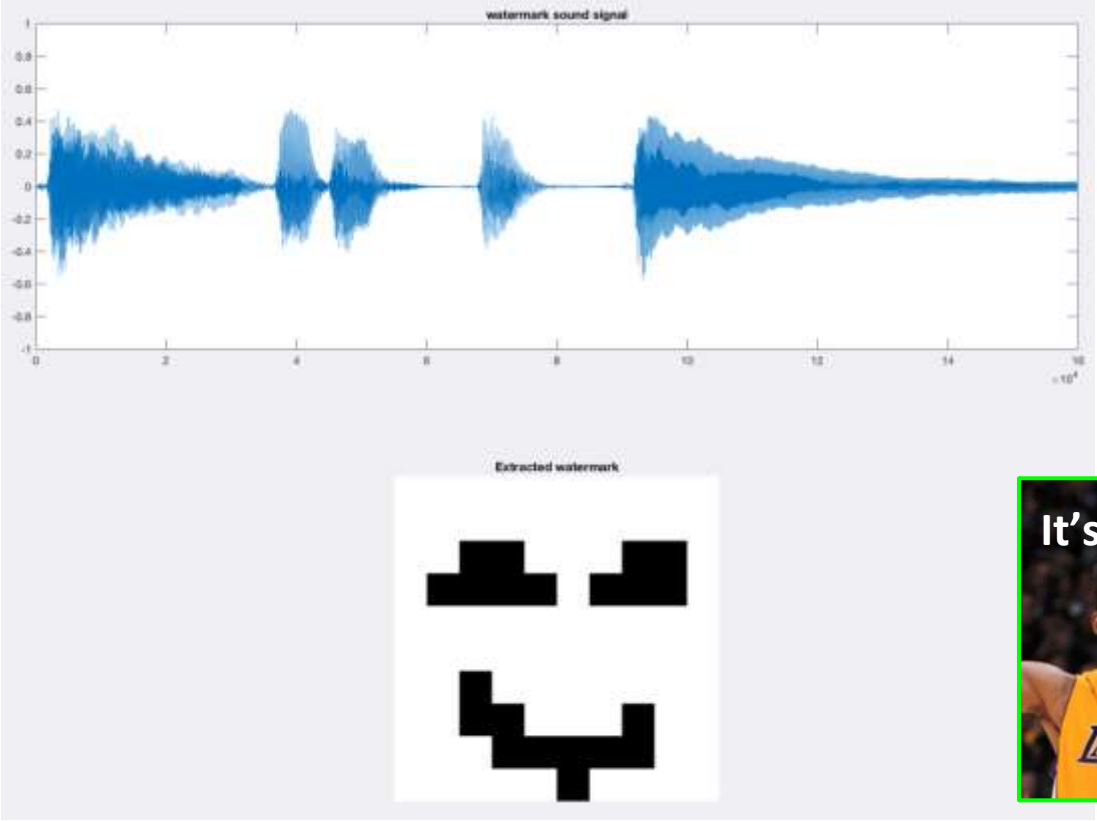
4. Improvement: 2D-1D sound watermark

- **2D to 1D? Yes:** M.A.T. Alsalami et al, “Digital Audio Watermarking: Survey”, 2003



4. Improvement: 2D-1D sound watermark

- **2D to 1D? Yes:** M.A.T. Alsalami et al, “Digital Audio Watermarking: Survey”, 2003



4. Improvement: 2D-1D sound watermark

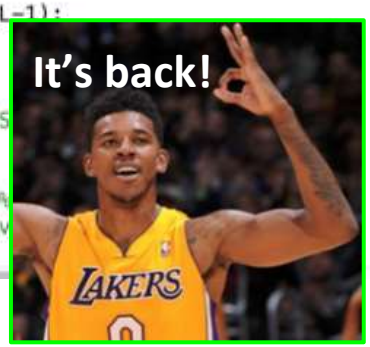
- **2D to 1D? Yes:** code implemented with DCT

Editor - /Users/lin/Desktop/IMproject 2/TianLin Yang/DWT-DCT-SVD-1Dsound/DwtDctSvd1DWatermark.m

DCTEncodeDecode.m | DCTLPwatermarking.m | DCTLPwatermarkingRGB.m | DCTLPwatermarkingRGBAttack.m | DwtDctSvd1DWatermark.m | DwtDctSvd1DExtract.m

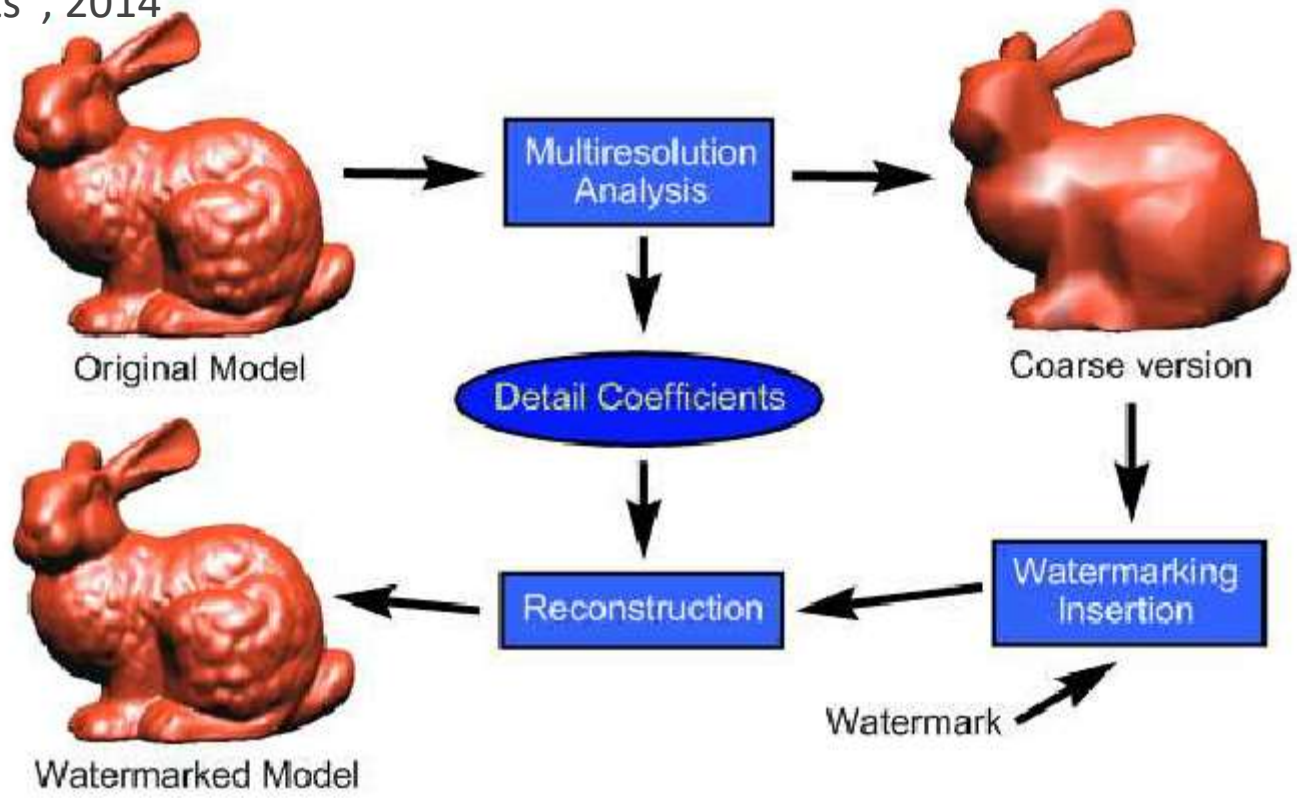
```
1   %Load original sound
2   [audio,fs]=audioread('OriginalSound.wav');
3   A=audio(1:160000);
4   AL=length(A);
5   %Drawing the original sound:
6   subplot(312);plot(A);
7   title('Original Sound');
8
9   %Watermarking photo
10  I=imread('Watermark.png');
11  I=im2bw(I);
12  subplot(311);imshow(I);
13  title('Watermarking image');
14  [m,n]=size(I);
15  %Reduce 2D to 1D of watermark
16  piexnum=1;
17  for i=1:m
18      for j=1:n
19          w(piexnum,1)=I(i,j);
20          piexnum=piexnum+1;
21      end
22  end
23  wl=size(w);
24
25  %Get distribute of 1D sound signal:
```

```
27  et the low-frequency(Higher energy) and high-freq(lower e
28  ica2=appcoef(cOri,lOri,'haar',2);
29  icd2=detcoef(cOri,lOri,2);
30  icd1=detcoef(cOri,lOri,1);
31  ica2L=length(Orica2);
32
33  CT Transfor
34  ica2DCT=dct(Orica2);
35  ivided to piece by piece
36  um=wl(1);   %Num of pieces
37  eceL=Orica2L/knum;  %ca2 every piece length
38  1;
39  lta=0.5;
40  dd watermark in to sound piece by piece
41  r i=1:knum
42      ca22=Orica2DCT(j:j+PieceL-1);
43      Y=ca22(1:PieceL/4);
44      Y=reshape(Y,10,10);
45
46      [U,S,V]=svd(Y);        %S
47      S1=S(1,1);
48      S2=S(2,2);
49      D=floor(S1/(S2*delta)); %
50      %Depend on D is odd or ev
51      if(mod(D,2)==0)
```



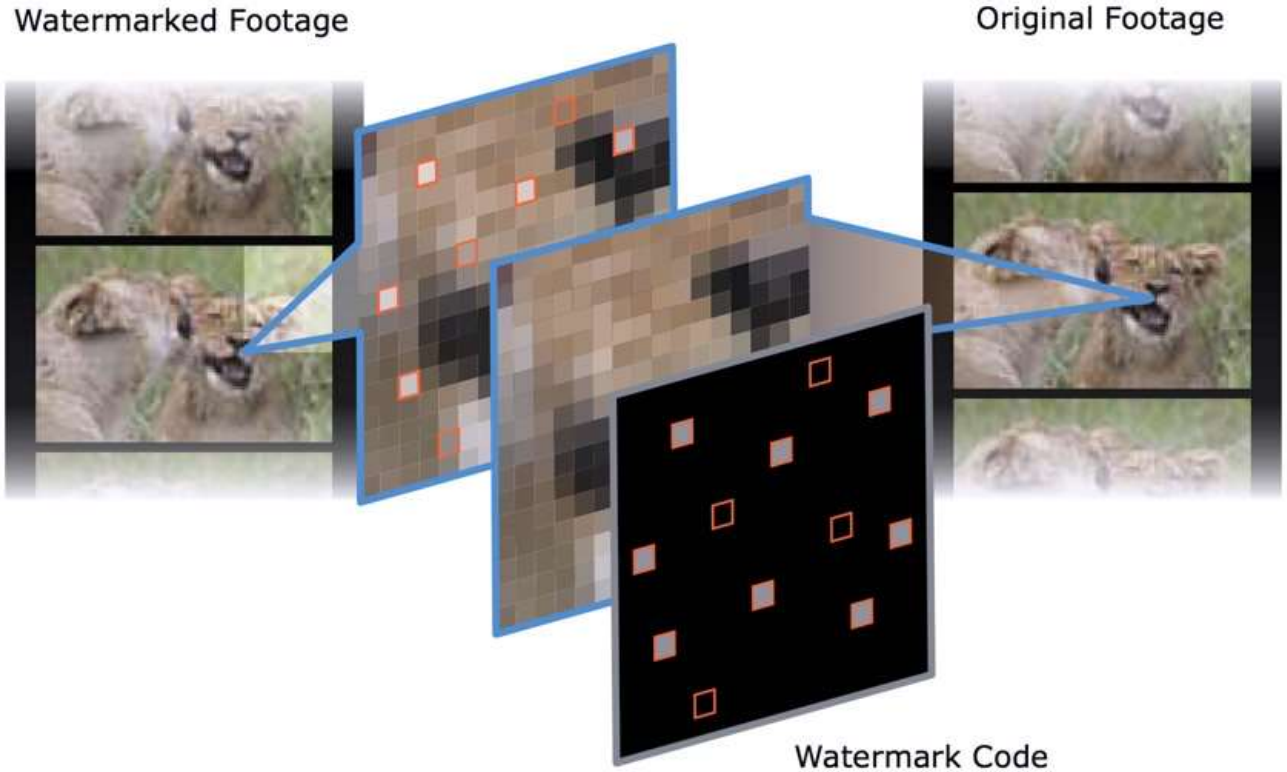
4. Improvement: 2D-3D research result

- **2D to 3D? Yes:** M. Corsini et al, "3D watermarking technology Visual quality aspects", 2014



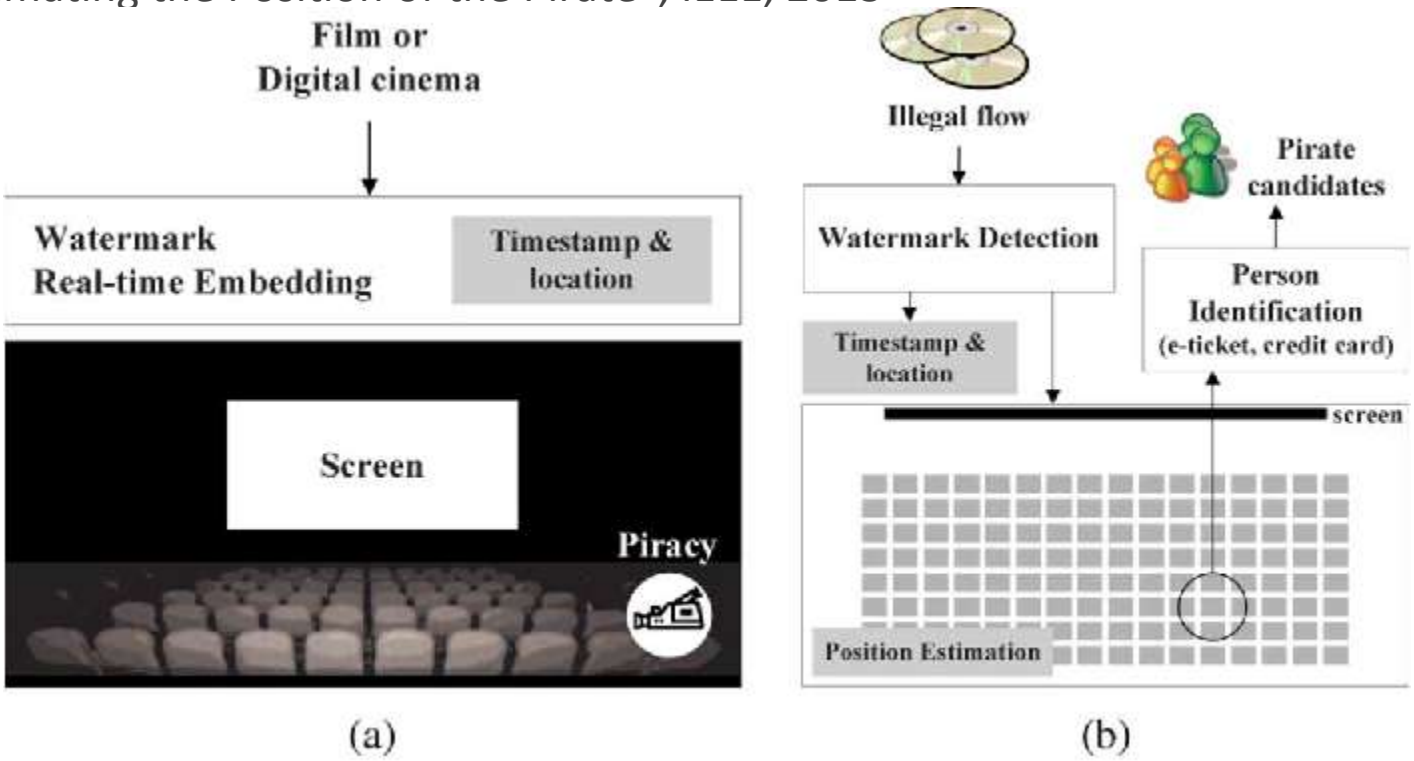
4. Improvement: Image to movie

- **Image to movie? Yes:** D. Milano ,“Content Control: Digital Watermarking and Fingerprinting”, RHOZET



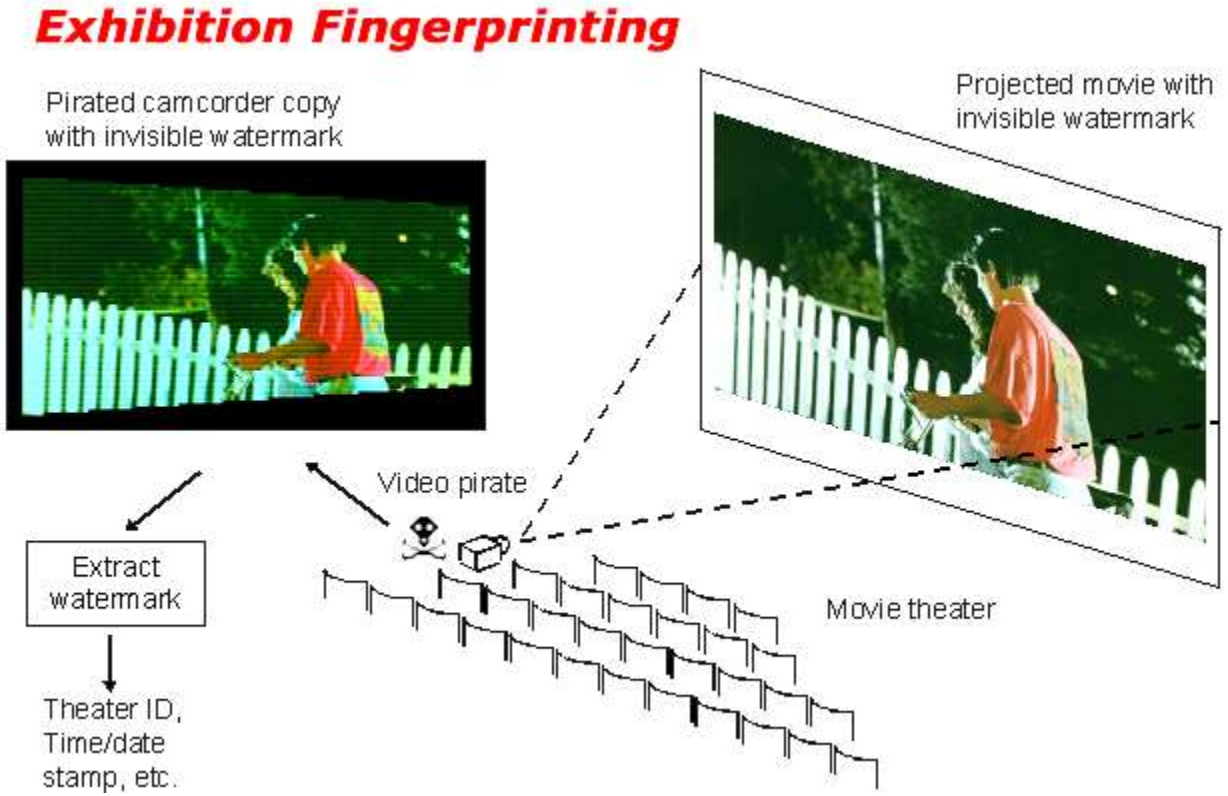
4. Improvement: Image to movie

- **Image to movie? Yes:** M.J. Lee et al,"Digital Cinema Watermarking for Estimating the Position of the Pirate", IEEE, 2013



4. Improvement: Image to movie

- **Image to movie? Yes:** M.J. Lee et al,"Digital Cinema Watermarking for Estimating the Position of the Pirate", IEEE, 2013



5. Conclusion

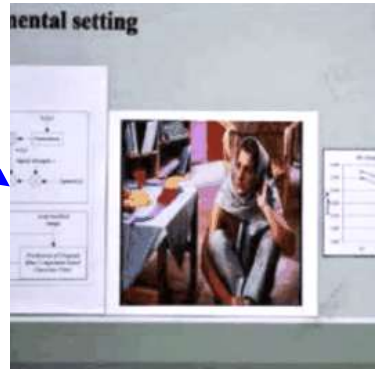
Our Paper

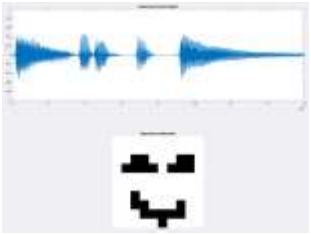
RGB vision

Attack simulation

'Key' feature

And, this





Sound-2D-3D-Movie

5. Conclusion

- **Our paper:**

  1. By implemented all the DCTPT code line-by-line, we get fully understand of this paper.

  2. By validate with their result, we ensure that our code is right.

  3. After research, we find this paper have some shortages.

- **Improvements:**

  1. To make the DCT for RGB photo, we designed RGB vision.

  2. To further checking the robustness, we did attack simulation.

  3. Dig out the solution for Anti-camera watermarking.

  4. Applied an Encode-Decode process, make ‘key’ feature available.

  5. Expand our research to 1D-2D-3D-Movie watermarking.

Reference

[1] J.P. Maheshwari et al, "Robust digital image watermarking using DCT based pyramid transform via image compression" IEEE ICCSP, 2015,

[2] M. Corsini et al, “3D watermarking technology Visual quality aspects”, Research Gate, 2014

[3] Peining Tao et al, “A robust multiple watermarking scheme in the Discrete Wavelet Transform domain”,Proceedings of the SPIE, Volume 5601, p. 133-144, 2004

[4] L.K. Saini, V. Shrivastava,“Analysis of Attacks on Hybrid DWT-DCT Algorithm for Digital Image Watermarking With MATLAB”, UCST – Volume 2 Issue 3, May-Jun 2014

[5] A. Pramila et al,“Camera based watermark extraction – problems and examples”, 2007

[6] K Thongkor et al, “Digital watermarking for camera-captured images based on just noticeable distortion and Wiener filtering”,Journal of Visual Communication and Image Representation VL 53, 2018

[7] L.P Feng et al,“A DWT-DCT Based Blind Watermarking Algorithm for Copyright Protection”, 2010 3rd International Conference on Computer Science and Information Technology, 2010

[8] M. Kumar et al,”Digital Image Watermarking using Fractional Fourier transform via image compression”, 2013 IEEE International Conference on Computational Intelligence and Computing Research, 2013

Reference

[9] I. Djurovic et al,“Digital watermarking in the fractional Fourier transformation domain”, Journal of Network and Computer Applications (2001) 24, 167 – 173, 2001

[10] J. Nin et al,“Digital Watermarking Techniques and Security Issues”, 2013 27th International Conference on Advanced Information Networking and Applications Workshops, 2013

[11] A. Tiwari et al,“Digital Image Watermarking using Fractional Fourier Transform with Different Attacks”, International Journal of Scientific Engineering and Technology (ISSN : 2277-1581) Volume No.3 Issue No.8, pp : 1008-1011, 2014

[12] V. P. S. Naidu ,“Novel Image Fusion Techniques using DCT”, ISSN: 1694-2108 | Vol. 5, No. 1. Sep 2013

[13] Z. J. XU et al,“Research on Image Watermarking Algorithm based on DCT”, Procedia Environmental Sciences 10 (2011) 1129 – 1135, 2011

[14] D. Milano ,“Content Control: Digital Watermarking and Fingerprinting”, RHOZET

[15] H. Tao et al,“Robust Image Watermarking Theories and Techniques: A Review”, Journal of Applied Research and Technology Vol.12 , 2014

[16] R.K. Sharma et al,“Practical Challenges for Digital Watermarking Applications”, EURASIP Journal on Applied Signal Processing 2002:2, 133–139 ©, 2002

Reference

[17] M.J. Lee et al,“Digital Cinema Watermarking for Estimating the Position of the Pirate”, IEEE, 2013

[18] M.A.T. Alsalami et al, “Digital Audio Watermarking: Survey”, 2003

Thank you for your attention!

UNIVERSITÉ
Concordia
UNIVERSITY

www.concordia.ca