DIGITAL IMAGE PROCESSING

FINAL PROJECT

Adel Bordbari

(*********)

Prof DR. *****

WINTER - 1401

TABLE OF CONTENTS

I.

INTRODUCTION

An overview of the final project, requirements, motives and goals

INTRODUCTION

Includes two main tasks:

compression and watermarking

Highlights the applications of DWT

Compression

- “Art and science of reducing the amount of required data for representation”
- Exists because data can be redundant and/or unnecessary
- Useful since a lot of data exists!
- Less requirements to save, share and process
- Can be lossy or lossless

INTRODUCTION

Watermarking

- Adding extra data for a purpose
- Somehow the opposite of compression
- Useful since a lot of data exists, again!
- Used for authentication and copyright issues

INTRODUCTION

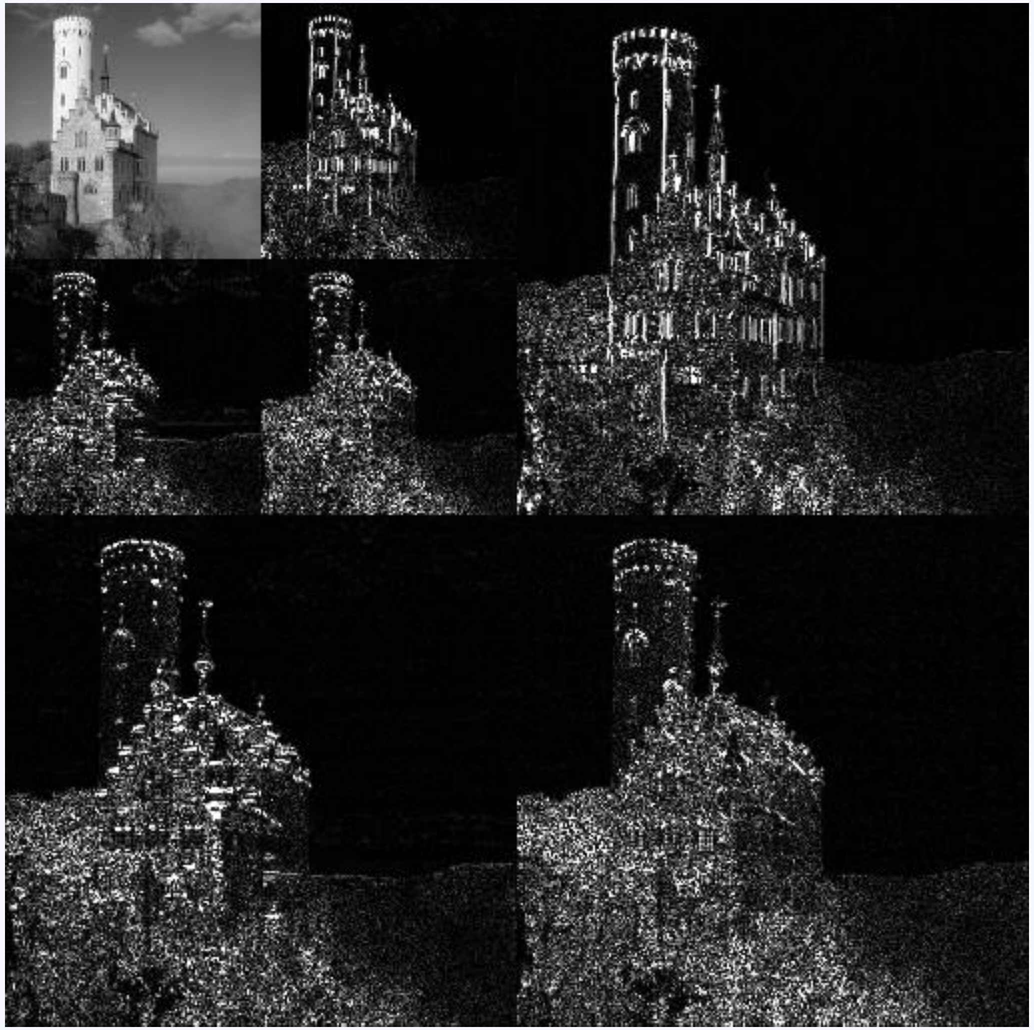
DWT

- Both tasks require DWT
- Discrete wavelet transform
- Sampling different wavelets separately
- Works like FFT; but better because considers temporal resolution (freq. and location)
- Based on mother wavelets: Haar, Daubechies, etc.
- Used in many methods: JPEG2000, JPEG XS, DjVu,

INTRODUCTION

Example:

coefficient matrices for2-D, 2-level DWT



II.

COMPRESSION

Thoughts, results and details on the first main task: Image compression using DWT

HIGHLIGHTS

Method

- Read 3 images
- Use 2-D DWT (Haar and Daubechies) to compress
- Quantize coefficients (local/global)
- Compare original and compressed images using MSE and PSNR metrics
- Show results in every step

Challenges

- Input images need to be diverse, include different frequency bands
- Some representations might be difficult to distinguish on computer screens
- Setting global threshold

Notes

- Images are in 1:1 ratio (square)
- Two images are small and mostly include lower frequencies
- One real life image, large, includes many frequencies and edges
- All color images, but processed in grayscale

Steps

1. Haar 3-L
    1.1. with local thresholding
2. Daubechies 3-L
    2.1. with local thresholding
3. Haar 5-L
    3.1. with local thresholding
4. Daubechies 5-L
    4.1. with local thresholding
5. Results

Image 1

“sculpture”

500x500px

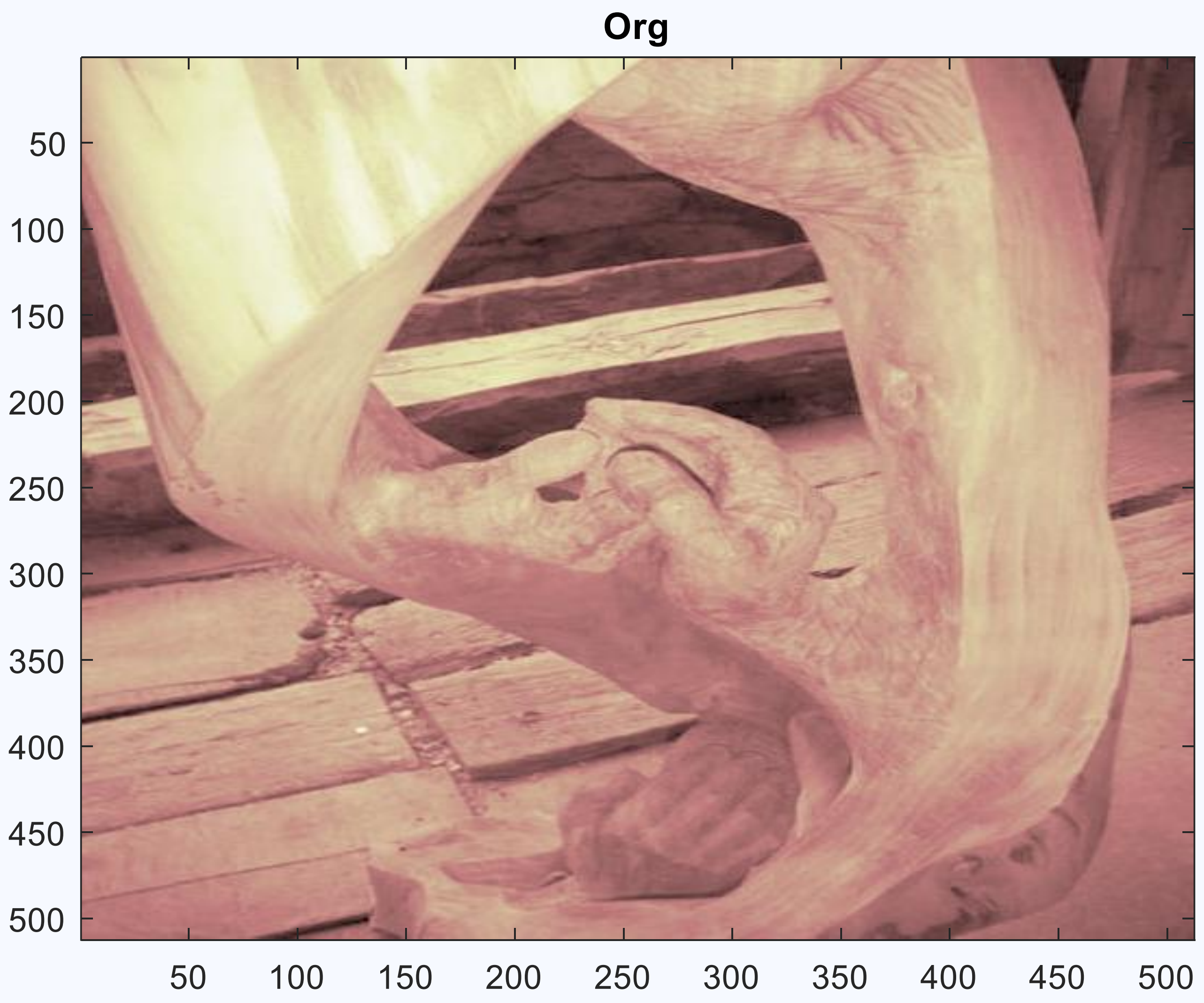


Image 2

“woman”

250x250px

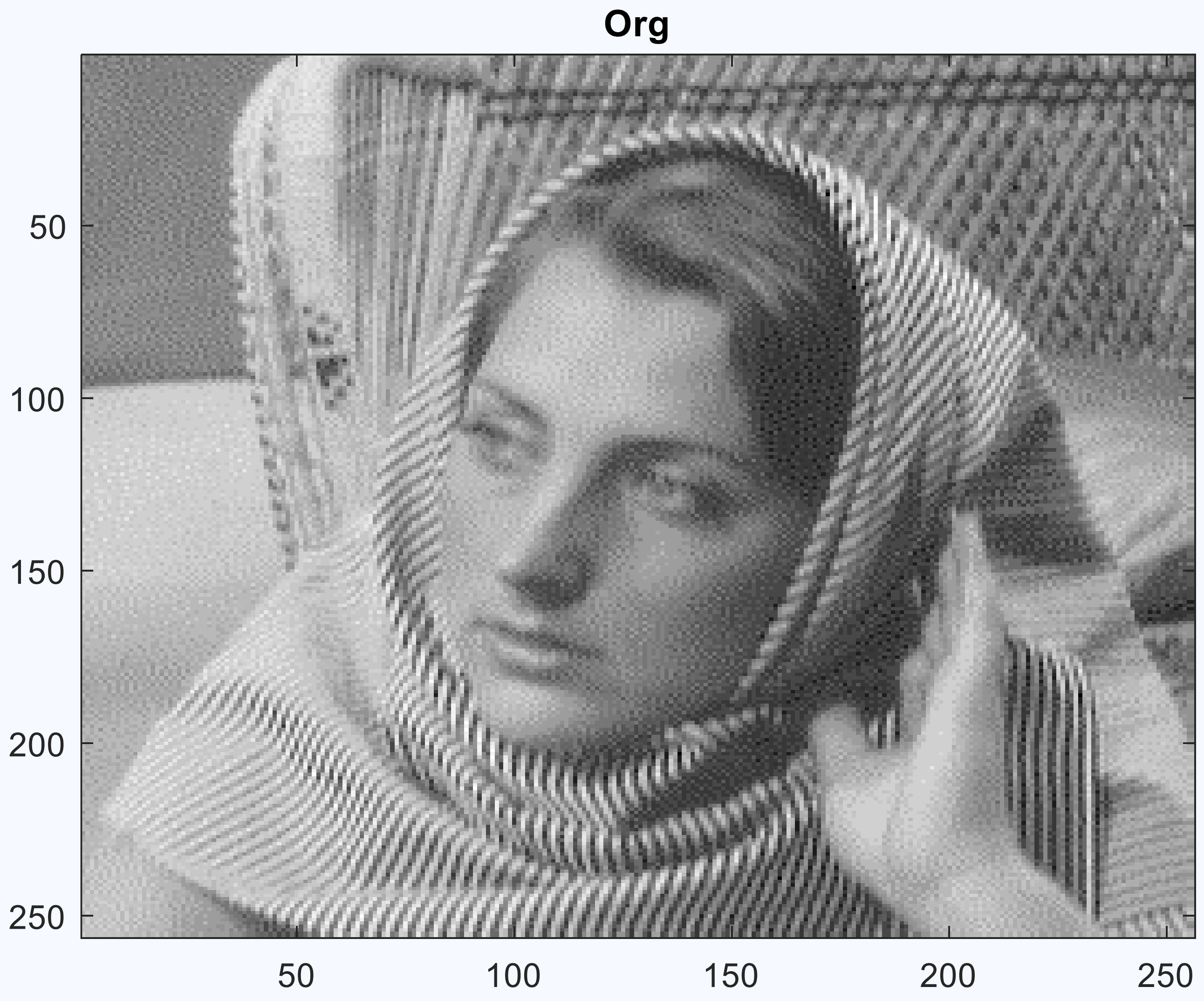
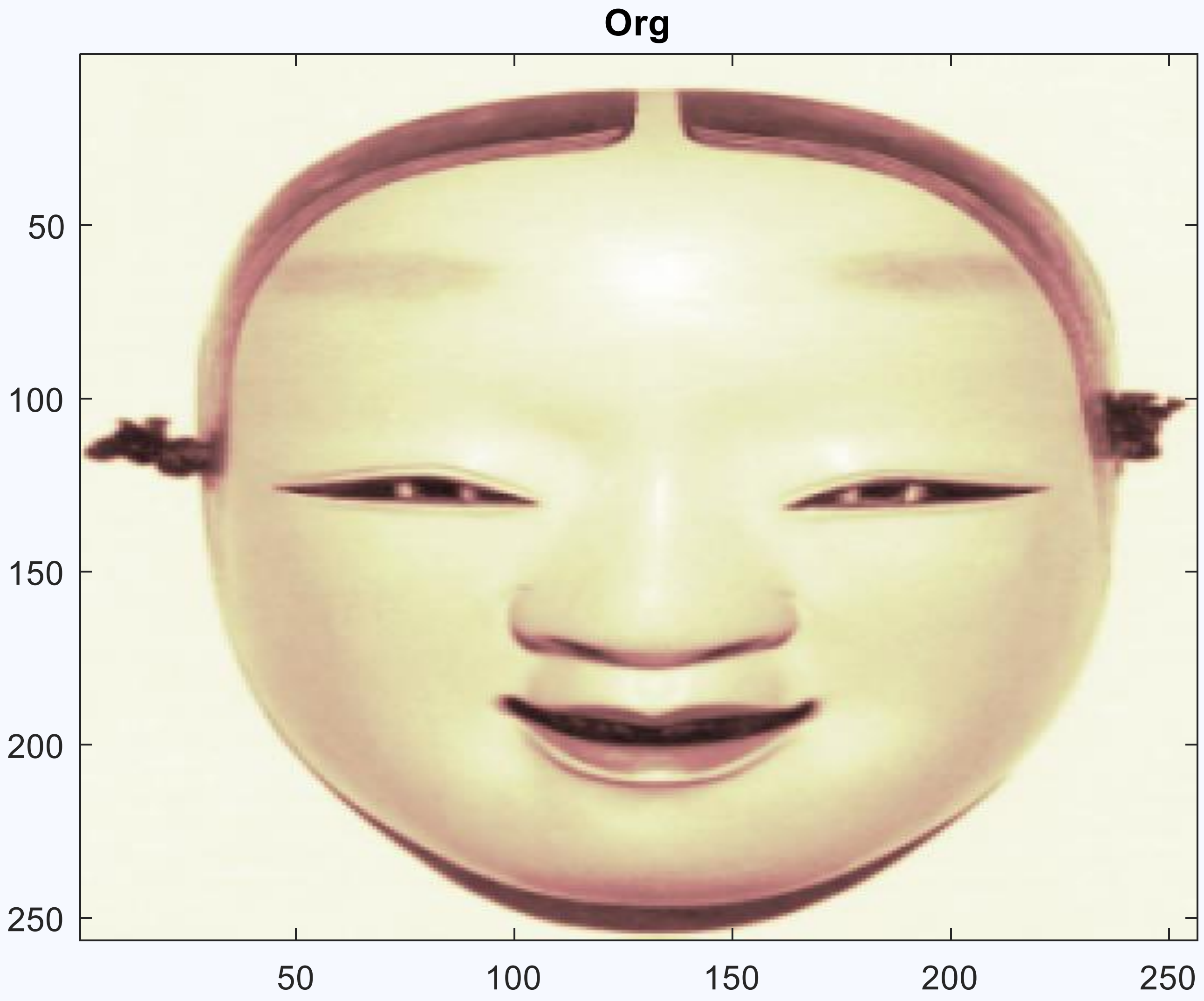


Image 3

“mask”

250x250px



II. 1

HAAR 3-L

1.1 WITH LOCAL THRESHOLDING

Compression ratio: **3.59**

MSE: **41.08**

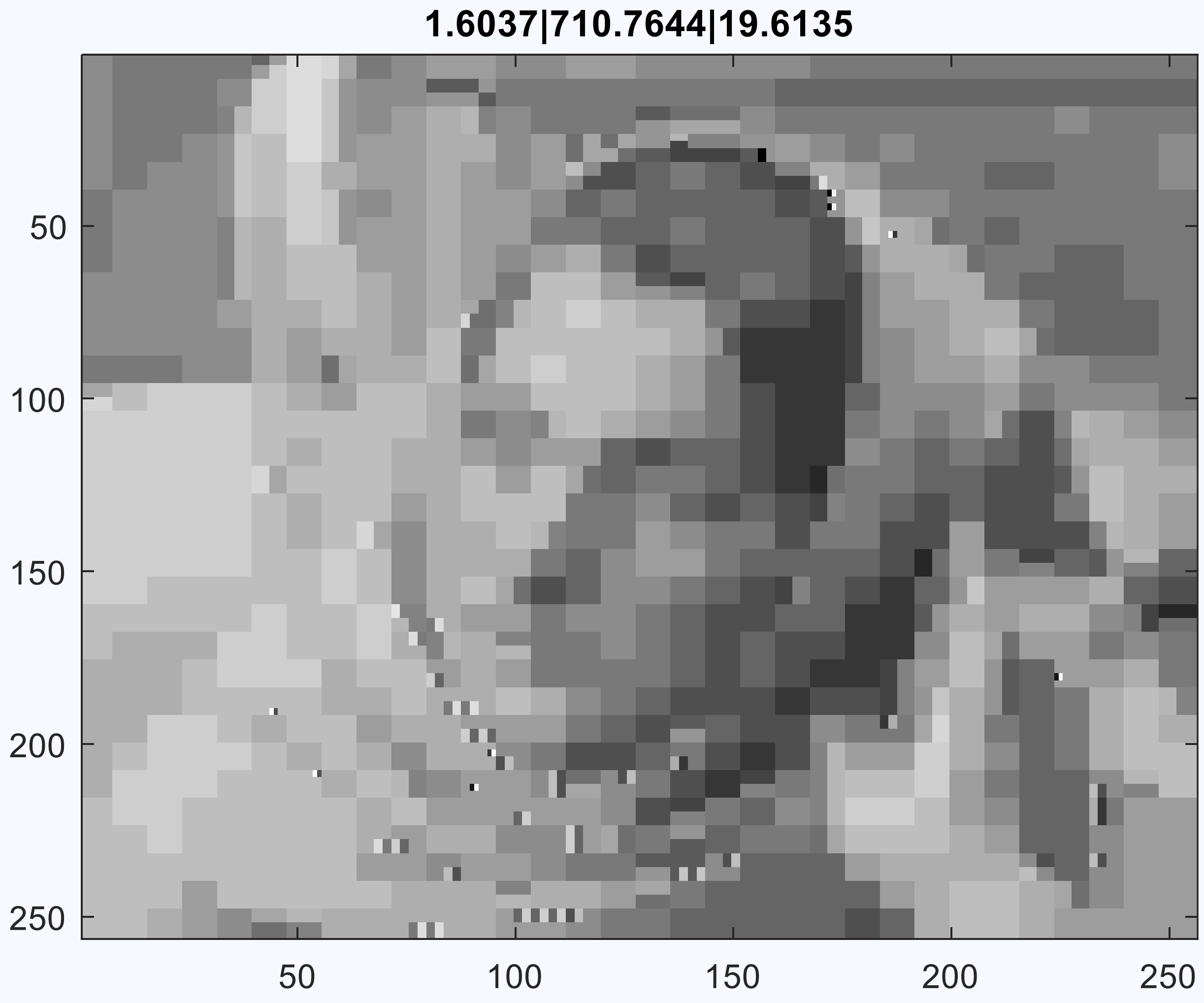
PSNR: **31.99**

1.2 WITH LOCAL THRESHOLDING

Compression ratio: **1.60**

MSE: **710.76**

PSNR: **19.61**

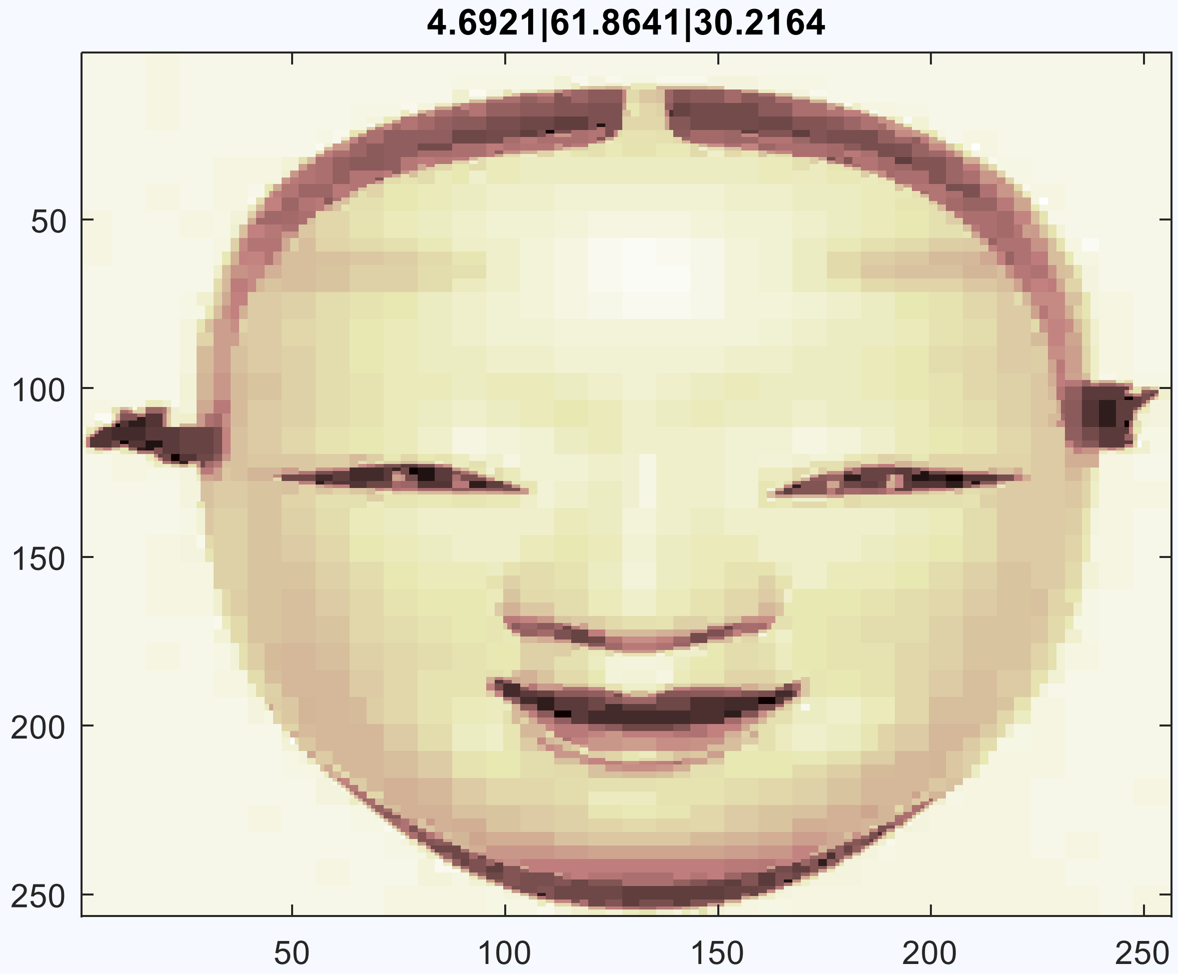


1.3 WITH LOCAL THRESHOLDING

Compression ratio: **4.69**

MSE: **61.86**

PSNR: **30.21**



II. 2

DAUBECHIES 3-L

2.1 WITH LOCAL THRESHOLDING

Compression ratio: **4.01**

MSE: **27.97**

PSNR: **33.66**

2.2 WITH LOCAL THRESHOLDING

Compression ratio: **2.08**

MSE: **631.78**

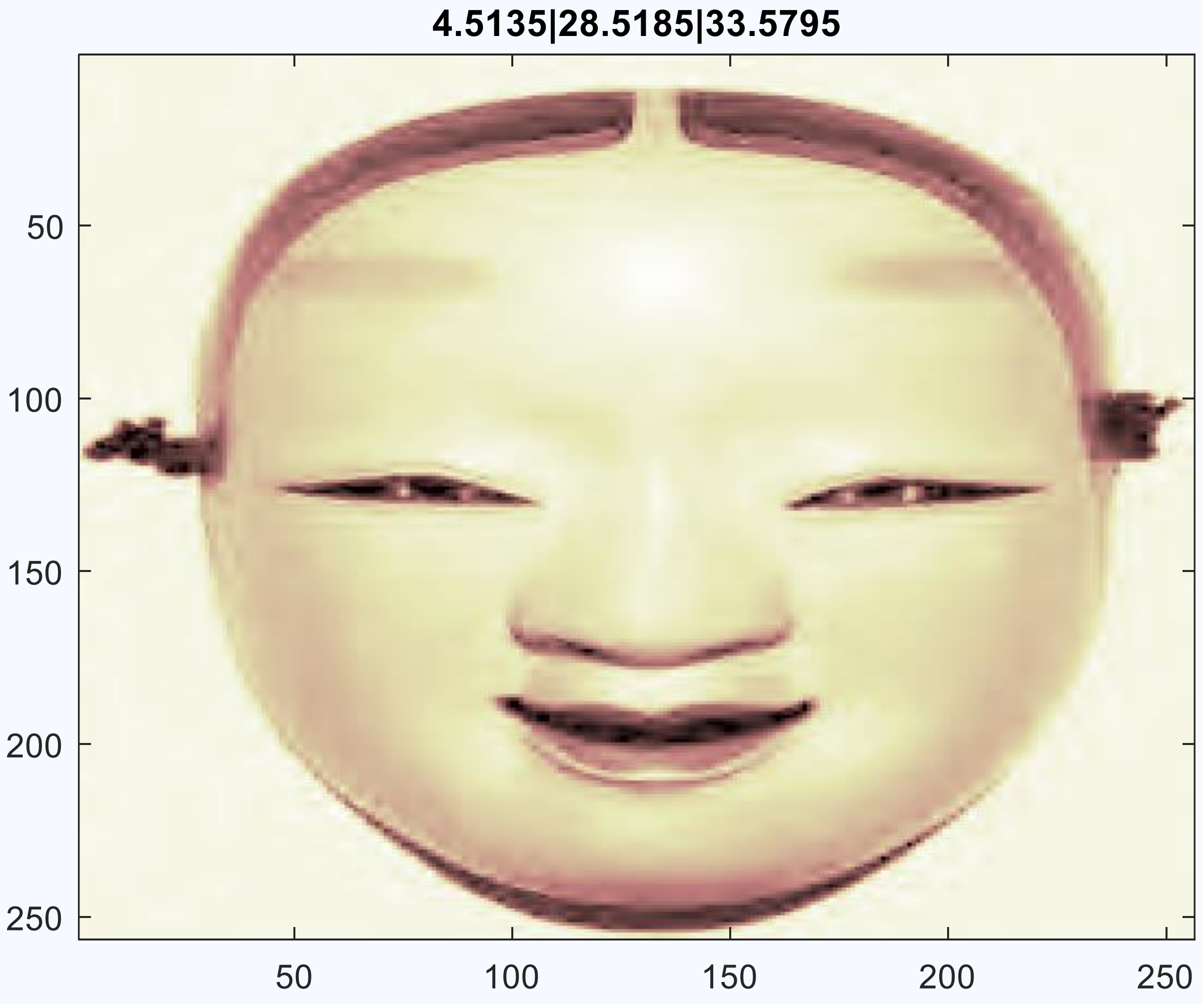
PSNR: **20.12**

2.3 WITH LOCAL THRESHOLDING

Compression ratio: **4.51**

MSE: **28.51**

PSNR: **33.57**



II. 3

HAAR 5-L

3.1. WITH LOCAL THRESHOLDING

Compression ratio: **3.06**

MSE: **41.70**

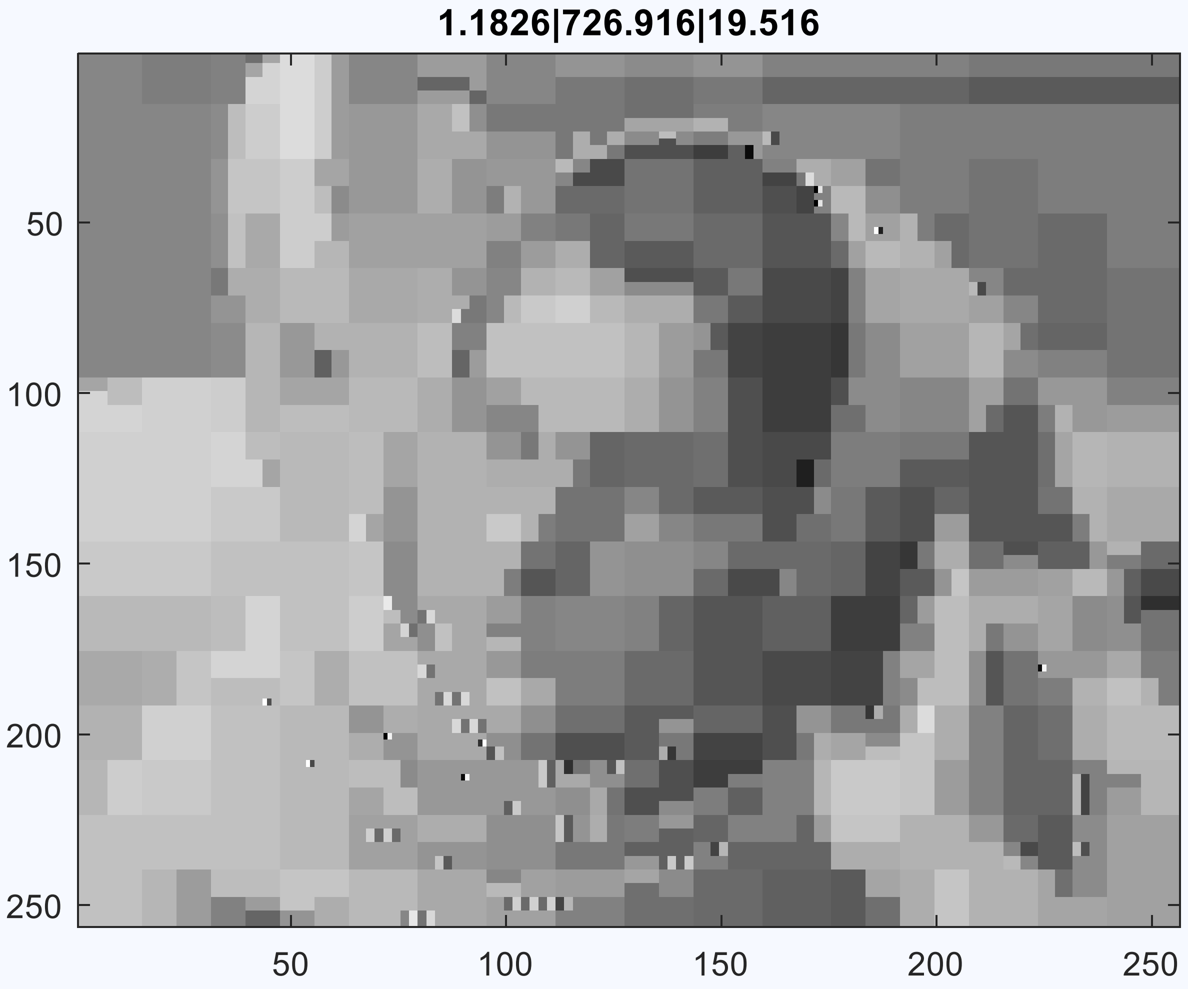
PSNR: **31.92**

3.2. WITH LOCAL THRESHOLDING

Compression ratio: **1.18**

MSE: **726.91**

PSNR: **19.51**

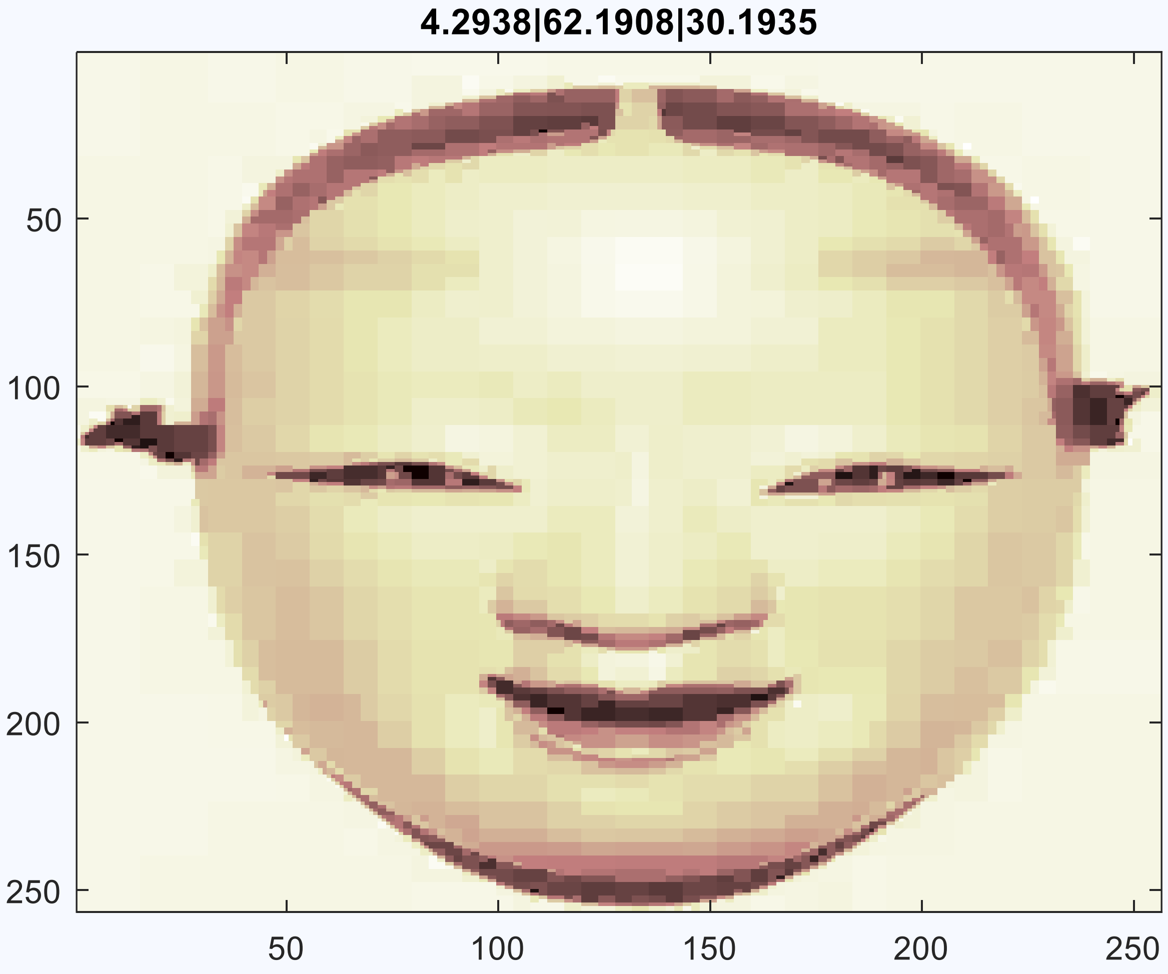


3.3. WITH LOCAL THRESHOLDING

Compression ratio: **4.29**

MSE: **62.19**

PSNR: **30.19**



II. 4

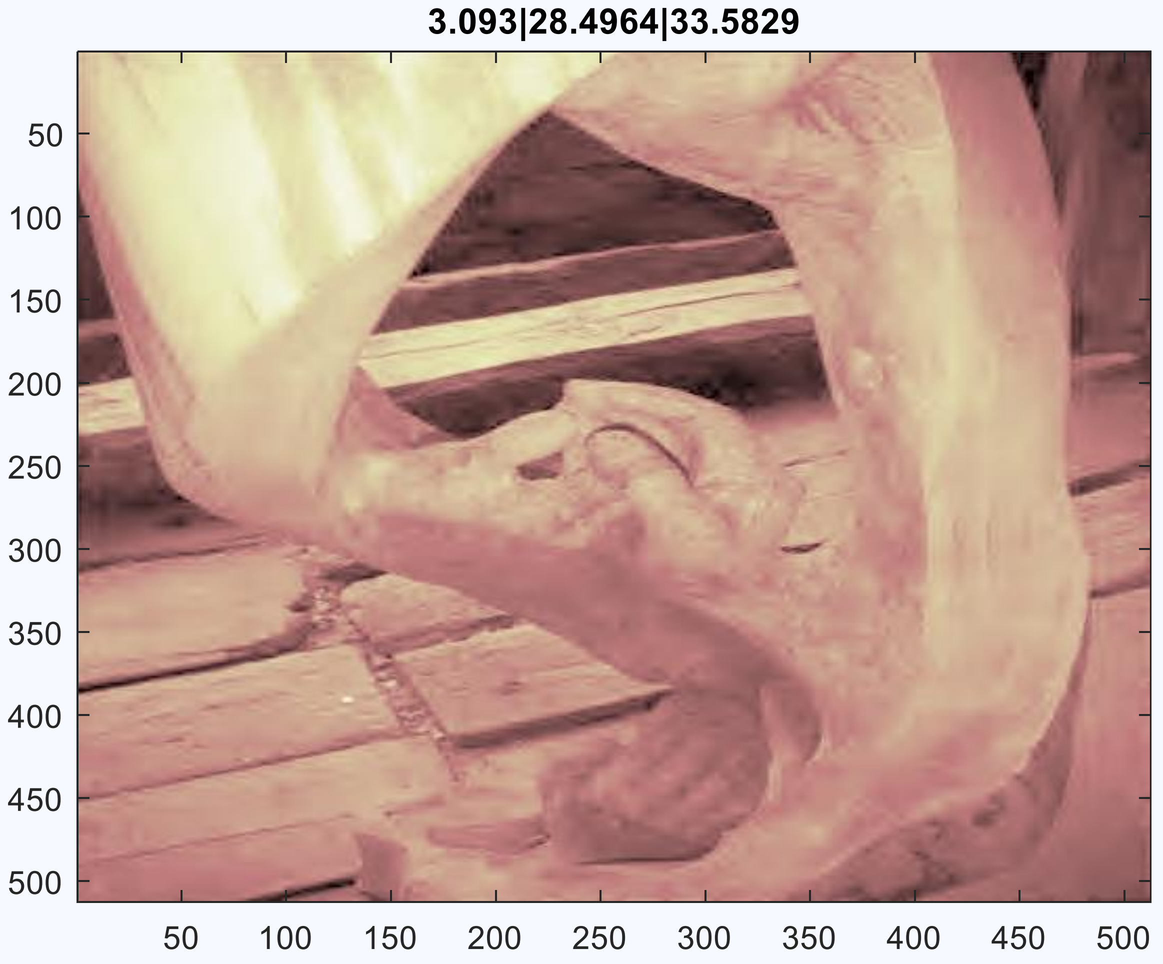
DAUBECHIES 5-L

4.1. WITH LOCAL THRESHOLDING

Compression ratio: **3.09**

MSE: **28.49**

PSNR: **33.58**



4.1. WITH LOCAL THRESHOLDING

Compression ratio: **1.30**

MSE: **646.95**

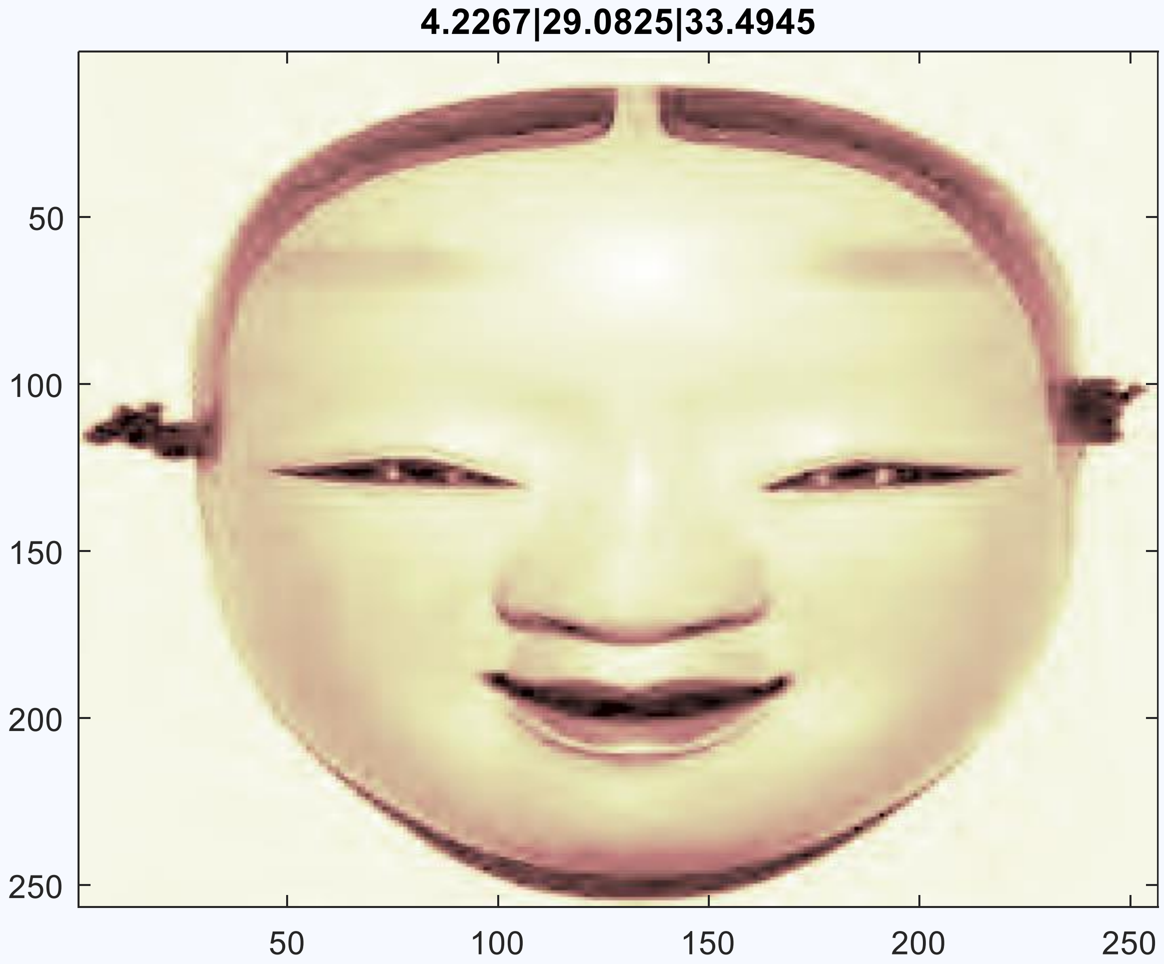
PSNR: **20.02**

4.1. WITH LOCAL THRESHOLDING

Compression ratio: **4.22**

MSE: **29.08**

PSNR: **33.49**



II. 5

CUMULATIVE RESULTS

Sculpture	H-3	DB-3	H-5	DB-5
C. ratio	3.59	4.01	3.06	3.09
MSE	41.08	27.97	41.7	28.49
PSNR	31.99	33.66	31.92	33.58
Woman	H-3	DB-3	H-5	DB-5
C. ratio	1.6	2.08	1.18	1.3
MSE	710.76	631.78	726.91	646.95
PSNR	19.61	20.12	19.51	20.02
Mask	H-3	DB-3	H-5	DB-5
C. ratio	4.69	4.51	4.29	4.22
MSE	61.86	28.51	62.19	29.08
PSNR	30.21	33.57	30.19	33.49

III.

WATERMARKING

Thoughts, results and details on the second main task: Image watermarking using DWT

HIGHLIGHTS

Method

- Read base and watermark images
- Use 2-D DWT (Haar and Daubechies) to decompose images
- Recompose watermark and base images together
- Save result
- De-watermark the result from previous step

Challenges

- Proper Q-K coefficients pair must be found experimentally
- Watermark visibility rate
- Tile or upscale watermark? what if base image is smaller?
- Some representations might be difficult to distinguish on computer screens

Notes

- Images are in 1:1 ratio (square)
- Watermark image is small and mostly include lower frequencies (20kB, 100x100px)
- Base image also mostly includes lower frequencies (78kB, 576x576px)
- Both color images, but processed in grayscale

Steps

1. Haar 3-L
   1.1. (k, q) = (0.2, 0.009)
   1.2. (k, q) = (0.6, 0.009)
   1.3. (k, q) = (1, 0.009)
   1.4. (k, q) = (1.4, 0.009)
   1.5. (k, q) = (1.8, 0.009)
   1.6. (k, q) = (0.2, 0.01)
   1.7. (k, q) = (0.6, 0.01)
   1.8. (k, q) = (1, 0.01)
   1.9. (k, q) = (1.4, 0.01)
   1.10. (k, q) = (1.8, 0.01)
2. Haar 3-L
   2.1. (k, q) = (0.2, 0.009)
   2.2. (k, q) = (0.6, 0.009)
   2.3. (k, q) = (1, 0.009)
   2.4. (k, q) = (1.4, 0.009)
   2.5. (k, q) = (1.8, 0.009)
   2.6. (k, q) = (0.2, 0.01)
   2.7. (k, q) = (0.6, 0.01)
   2.8. (k, q) = (1, 0.01)
   2.9. (k, q) = (1.4, 0.01)
   2.10. (k, q) = (1.8, 0.01)
3. Results

Base Image

“disco”

576x576px

Watermark Image

“circle”

100x100px

III. 1

HAAR 3-L

1.1. (0.2, 0.009)

8928.6798|8.6229

MSE: **8928**

PSNR: **8.62**

Size: **14.2**

1.2. (0.6, 0.009)

MSE: **2174**

PSNR: **14.75**

Size: **24.2**

2174.9373|14.7563

1.3. (1, 0.009)

MSE: **21.48**

PSNR: **34.80**

Size: **32.5**

21.4871|34.809

1.4. (1.4, 0.009)

MSE: **1968**

PSNR: **15.18**

Size: **38.1**

1968.9975|15.1884

1.5. (1.8, 0.009)

MSE: **4380**

PSNR: **11.71**

Size: **39.1**

4380.0609|11.716

1.6. (0.2, 0.01)

MSE: **8905**

PSNR: **8.63**

Size: **14.2**

8905.0808|8.6344

1.7. (0.6, 0.01)

MSE: **2152**

PSNR: **14.8**

Size: **24.1**

2152.0931|14.8022

1.8. (1, 0.01)

MSE: **21.88**

PSNR: **34.72**

Size: **32.6**

21.8871|34.7289

1.9. (1.4, 0.01)

MSE: **1981**

PSNR: **15.16**

Size: **38.1**

1981.0212|15.1619

1.10. (1.8, 0.01)

MSE: **4393**

PSNR: **11.70**

Size: **39.5**

4393.7056|11.7025

III. 2

DAUBECHIES 3-L

2.1. (0.2, 0.009)

MSE: **8928**

PSNR: **8.62**

Size: **14.2**

8928.6939|8.6229

2.2. (0.6, 0.009)

MSE: **2174**

PSNR: **14.75**

Size: **24.2**

2174.9351|14.7563

2.3. (1, 0.009)

MSE: **21.48**

PSNR: **34.80**

Size: **32.5**

21.4871|34.809

2.4. (1.4, 0.009)

MSE: **1968**

PSNR: **15.18**

Size: **38.1**

1969|15.1883

2.5. (1.8, 0.009)

MSE: **4380**

PSNR: **11.71**

Size: **39.1**

4379.989|11.7161

8905.6481|8.6341

2.6. (0.2, 0.01)

MSE: **8905**

PSNR: **8.63**

Size: **14.2**

2.7. (0.6, 0.01)

MSE: **2152**

PSNR: **14.8**

Size: **24.1**

2152.1038|14.8022

2.8. (1, 0.01)

MSE: **21.85**

PSNR: **34.73**

Size: **32.6**

21.8562|34.735

2.9. (1.4, 0.01)

MSE: **1981**

PSNR: **15.16**

Size: **38.1**

1980.771|15.1625

2.10. (1.8, 0.01)

MSE: **4393**

PSNR: **11.70**

Size: **39.6**

4393.3549|11.7028

III. 3

CUMULATIVE RESULTS

H-3	MSE	PSNR	SIZE
0.2, 0.009	**8928**	**8.62**	**14.2**
0.6, 0.009	**2174**	**14.75**	**24.2**
1.0, 0.009	**21.48**	**34.80**	**32.5**
1.4, 0.009	**1968**	**15.18**	**38.1**
1.8, 0.009	**4380**	**11.71**	**39.1**
0.2, 0.01	**8905**	**8.63**	**14.2**
0.6, 0.01	**2152**	**14.8**	**24.1**
1.0, 0.01	**21.88**	**34.72**	**32.6**
1.4, 0.01	**1981**	**15.16**	**38.1**
1.8, 0.01	**4393**	**11.70**	**39.5**

DB-3	MSE	PSNR	SIZE
0.2, 0.009	**8928**	**8.62**	**14.2**
0.6, 0.009	**2174**	**14.75**	**24.2**
1.0, 0.009	**21.48**	**34.80**	**32.5**
1.4, 0.009	**1968**	**15.18**	**38.1**
1.8, 0.009	**4380**	**11.71**	**39.1**
0.2, 0.01	**8905**	**8.63**	**14.2**
0.6, 0.01	**2152**	**14.8**	**24.1**
1.0, 0.01	**21.85**	**34.73**	**32.6**
1.4, 0.01	**1981**	**15.16**	**38.1**
1.8, 0.01	**4393**	**11.70**	**39.6**

IV.

CODE

Explaining the codes used on this project: functions, procedures, conversions and representations

4. CODE

Highlights

1. Overview
2. Compression
3. Watermarking
4. De-Watermarking

Notes

- Using Matlab (R2018b)
- Windows 10
- Official docs
- Images pasted as figure

4.1. Overview

Main functions

1. `dwt2()`: 2-D, 1-Level DWT on an input
2. `idwt2()`: 2-D, 1-Level IDWT on LL-LH-HL-HH frequency bands
3. `wavedec2()`: 2-D N-level wavelet decomposition on an input
4. `waverec2()`: 2-D, N-Level wavelet reconstruction on a decomposition structure
5. `wcompress()`: True un/compression of an image using method `compmthd`

4.2. Compression

```
load woman;
n = 5;
w = 'db3';
```

- Load image
- Image names: `sculpture woman mask`
- Set DWT level
- Set wavelet family

```
[cr,bpp] =
wcompress('c',X,'mask.wtc‘
        ,'lvl_mmc','wname',w
        ,'level',n);
```

- Compress image X
- Save compression structure as `'mask.wtc‘`
- `'c‘` for compression; `‘u‘` for un-compression.
- Process using set family-level provided above
- Save compression ratio and bit-per-pixel measurements

4.2. Compression

```
Xc =
wcompress('u','mask.wtc');
delete('mask.wtc')
```

- Save compressed image as Xc, using the structure from prev. step
- Delete structure file

```
X = double(X);
D = abs(X-Xc).^2;
mse  = sum(D(:))/numel(X);
psnr = 10*log10(255*255/mse);
```

- Convert input image to double
- Calculate MSE and PSNR

4.2. Compression

```
figure
image(Xc);
title(strcat(num2str(cr),'|',num2str(mse),'|',num2str(psnr)));
colormap(map);

figure
image(X);
title("Org");
colormap(map);
```

- Show output image + MSR/PSNR
- Show original image

4.3. Watermarking

```
base = imread('disco.jpg');
wm = imread('circle.png');

wv = 'db3';
lvl = 3;

k = 1.8; % 0.2 0.6 1 1.4 1.8
q = 0.01; % 0.009 0.01

watermark(base, wm, wv, k, q, lvl);
```

- Read local image as `base`
- Read watermark image as `wm`
- Set wavelet family as `wv`
  - Options: `db3`, `haar`
- Set wavelet level `lvl`
- Set base image coefficient k
- Set base image coefficient q
- Using set values, run the function `watermark`

4.3. Watermarking

```
function y=watermark(A,B,wave,k,q,lvl)
    host = A;
    wm = B;
    [m, n, p] = size(host);
    [h_c, h_s] = wavedec2(host, lvl, wave);
    wm = imresize(wm, [m n]);
    [w_c, w_s] = wavedec2(wm, lvl, wave);
    w_c = (k * h_c) + (q * w_c);
    wmRes = waverec2(w_c, h_c, wave);
```

- Define function `watermark` with 6 arguments
- Save base image as `host`
- Save watermark image as `wm`
- Get dimensions and color channels of host image
- Run wavelet decomposition on base image
- Resize watermark to match base image’s dimensions
- Run wavelet decomposition on base image
- Apply set coefficients
- Run wavelet reconstruction on watermark and base image’s LL bands, save as `wmRes`

4.3. Watermarking

```
imwrite(uint8(wmRes), 'watermarked.jpg');
compare = imread('watermarked.jpg');
compare = double(compare);
base = double(host);
D = abs(compare - base).^2;
mse  = sum(D(:))/numel(compare);
psnr = 10*log10(255*255/mse);
```

- Save the result in storage as `atermarked.jpg`
- Convert result to image of type double
- Calculate MSE/PSNR

4.3. Watermarking

```
    figure
    imshow(uint8(wmRes));
    deets = strcat(num2str(mse), '|', num2str(psnr));
    title(deets);
    y = 'Watermarked succesfully';
end
```

- Show image + MSE/PSNR measurements
- Return success message
- Quit function

4.4. De-Watermarking

```
base = imread('disco.jpg');
wm = imread('circle.png');
wmres = imread('watermarked.png');
wv = 'haar';
ext_watermark(base, wm, wmres, wv);
```

- Read base and watermark image
- Read result image
- Set wavelet family
- Run `ext_watermark` function, pass in set arguments

4.4. De-Watermarking

```
function y=ext_watermark(A,B,C,wave)
    host = A;
    [m, n, p] = size(host);
    [host_LL, host_LH, host_HL, host_HH] = dwt2(host ,wave);
```

- Define function with 4 arguments
- Pass in base, watermark and result images wavelet family
- Read base image as `host`
- Run 2-D 1-L DWT on base image
- Save 4 different frequency bands

4.4. De-Watermarking

```
water_mark = B;
water_mark = imresize(water_mark,[m n]);

[water_mark_LL, water_mark_L,
water_mark_HL, water_mark_HH] =
        dwt2(water_mark,wave);
```

- Read watermark image as B
- Resize watermark image to match base image
- Run 2-D 1-L DWT on watermark image
- Save all 4 frequency bands

4.4. De-Watermarking

```
wm = C;
[wm_LL, wm_LH, wm_HL, wm_HH] = dwt2(wm, wave);
rec_watermark = (wm_LL - host_LL) / 0.03;
ext = idwt2(rec_watermark, water_mark_LH,
        water_mark_HL, water_mark_HH, wave);
```

- Read result image as `C`
- Run 2-D 1-L DWT on result image
- Save all 4 frequency bands
- Run 2-D 1-L IDWT on result image's bands
- Save extracted watermark as `ext`

4.4. De-Watermarking

```
    figure
        subplot(1,2,1);
        imshow(uint8(ext));
        title('Extracted watermark');
        subplot(1,2,2);
        imshow(255 - uint8(ext));
        title('Negative watermark');
        y = 'De-watermarked successfully';
end
```

- Show extracted watermark
- Show extracted watermark with inverse color (sometimes easier to see)
- Return success message
- Quit function

V.

CONCLUSION

Out-takes on the overall procedures the project

5. CONCLUSION

Compression

- More DWT depth = Better results
- Reviewed families not much different
  - Others un-reviewed families might be different
- Images with low freq. = easier to compress
- Images with patterns and high freq. = hard to compress

5. CONCLUSION

Watermarking

- Different families = no difference AT ALL!
- Depth 5 was also reviewed, not mentioned in the report
  - Also no difference at all!
- Only (k,q) values lead to different results
- K=1 leads to best results
- Higher Q = bigger file size

VI.

FUTURE WORKS

What needs to be done, suggested revisions and extensions on method and functionality

6. FUTURE WORK

- Try other wavelet families: `bior`, `syml`, `spiht`, etc.
- Try higher DWT levels
- Try both global and local thresholding for coeff. Matrices
  - Using `ddencom`, `adaptthresh`, etc.
- Try larger image
- Try tiling the watermark
- Add exception catching for bad inputs

VI.

REFERENCES

Books, links, blogposts, etc.

6. REFERENCES

1. https://www.mathworks.com/help/wavelet/ref/wcompress.html
2. https://www.mathworks.com/help/wavelet/ref/wavedec2.html?searchHighlight=wavedec2&s_tid=srchtitle_wavedec2_1
3. https://www.mathworks.com/help/wavelet/ref/waverec2.html?searchHighlight=waverec2&s_tid=srchtitle_waverec2_1
4. https://www.mathworks.com/help/wavelet/ref/dwt2.html?searchHighlight=dwt2&s_tid=srchtitle_dwt2_1
5. https://www.mathworks.com/help/wavelet/ref/idwt2.html?searchHighlight=idwt2&s_tid=srchtitle_idwt2_1

6. REFERENCES

6. Discrete wavelet transform. (2022, December 15). In *Wikipedia*. https://en.wikipedia.org/wiki/Discrete_wavelet_transform
7. https://www.scribd.com/document/436856865/Concise-Introduction-to-Wavelets
8. S. Mallat, A Wavelet Tour of Signal Processing, 2nd ed. San Diego, CA: Academic, 1999
9. Gonzales, Rafael C., and Paul Wintz. *Digital image processing*. Addison-Wesley Longman Publishing Co., Inc., 2008.


THANK YOU