16.10.2023

Digital Image Processing

Image Compression - 2

Makarand Tapaswi

Center for Visual Information Technology (CVIT), IIIT Hyderabad

Most slides borrowed from Ravi Kiran @CVIT!

Diversion: Floating point standards

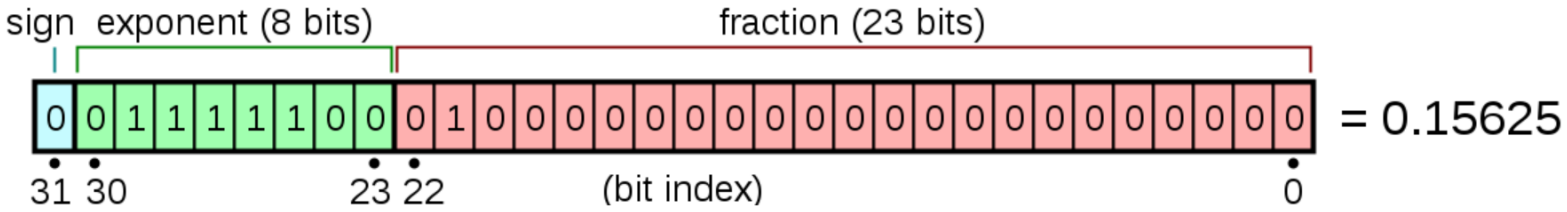
- Pick the number based on suitable exponent

- float32 (IEEE 754)
  - Sign bit: 1 bit
  - Exponent width: 8 bits
  - Significand precision: 24 bits (23 stored)

$$(-1)^{b_{31}} \times 2^{(b_{30}b_{29}\ldots b_{23})_2 - 127} \times (1.b_{22}b_{21}\ldots b_0)_2\,,$$

which yields

$$\text{value} = (-1)^{\text{sign}} \times 2^{(E-127)} \times \left(1 + \sum_{i=1}^{23} b_{23-i}2^{-i}\right).$$



LZW Coding
(addresses interpixel redundancy)

- Requires no prior knowledge of symbol probabilities.
- Assigns fixed length code words to variable length symbol sequences.
  - There is no one-to-one correspondence between source symbols and code words.
- Included in GIF, TIFF and PDF file formats

LZW Coding

- A codebook (or dictionary) needs to be constructed.

- Initially, the first 256 entries of the dictionary are assigned to the gray levels 0,1,2,..,255 (i.e., assuming 8 bits/pixel)

Consider a 4x4, 8 bit image

39	39	126	126
39	39	126	126
39	39	126	126
39	39	126	126

Initial Dictionary

Dictionary Location	Entry
0	0
1	1
.	.
255	255
256	-
511	-

LZW Coding (cont'd)

As the encoder examines image pixels, gray level sequences (i.e., blocks) that are not in the dictionary are assigned to a new entry.

39	39	126	126
39	39	126	126
39	39	126	126
39	39	126	126

Dictionary Location	Entry
0	0
1	1
.	.
255	255
256	39-39
511	-

- Is 39 in the dictionary……..Yes
- What about 39-39…….No
  * Add 39-39 at location 256

Example

39	39	126	126
39	39	126	126
39	39	126	126
39	39	126	126

CR = empty
repeat
P=next pixel
CS=CR + P

If CS is found:
(1) No Output
(2) CR=CS

else:
(1) Output D(CR)
(2) Add CS to D
(3) CR=P

Concatenated Sequence: CS = CR + P

(CR) (P)

Currently Recognized Sequence	Pixel Being Processed	Encoded Output	Dictionary Location (Code Word)	Dictionary Entry
	39			
39	39	39	256	39-39
39	126	39	257	39-126
126	126	126	258	126-126
126	39	126	259	126-39
39	39			
39-39	126	256	260	39-39-126
126	126			
126-126	39	258	261	126-126-39
39	39			
39-39	126			
39-39-126	126	260	262	39-39-126-126
126	39			
126-39	39	259	263	126-39-39
39	126			
39-126	126	257	264	39-126-126
126		126		

Decoding LZW

- Use the dictionary for decoding the “encoded output” sequence.
- The dictionary need not be sent with the encoded output.
- Can be built on the “fly” by the decoder as it reads the received code words.

Run-length coding (RLC)
(addresses interpixel redundancy)

- Reduce the size of a repeating string of symbols (i.e., runs):

1 1 1 1 1 0 0 0 0 0 0 1 → (1,5) (0, 6) (1, 1)

a a a b b b b b b c c → (a,3) (b, 6) (c, 2)

- Encodes a run of symbols into two bytes: **(symbol, count)**
- Can compress any type of data but cannot achieve high compression ratios compared to other compression methods.

Combining Huffman Coding with Run-length Coding

- Assuming that a message has been encoded using Huffman coding, additional compression can be achieved using run-length coding.

0 1 0 1 0 0 1 1 1 1 0 0

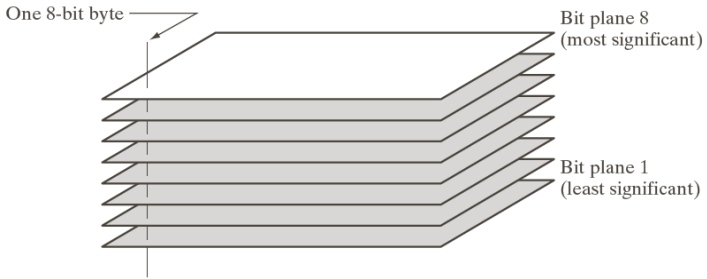
e.g., (0,1)(1,1)(0,1)(1,0)(0,2)(1,4)(0,2)

Bit-plane coding
(addresses interpixel redundancy)

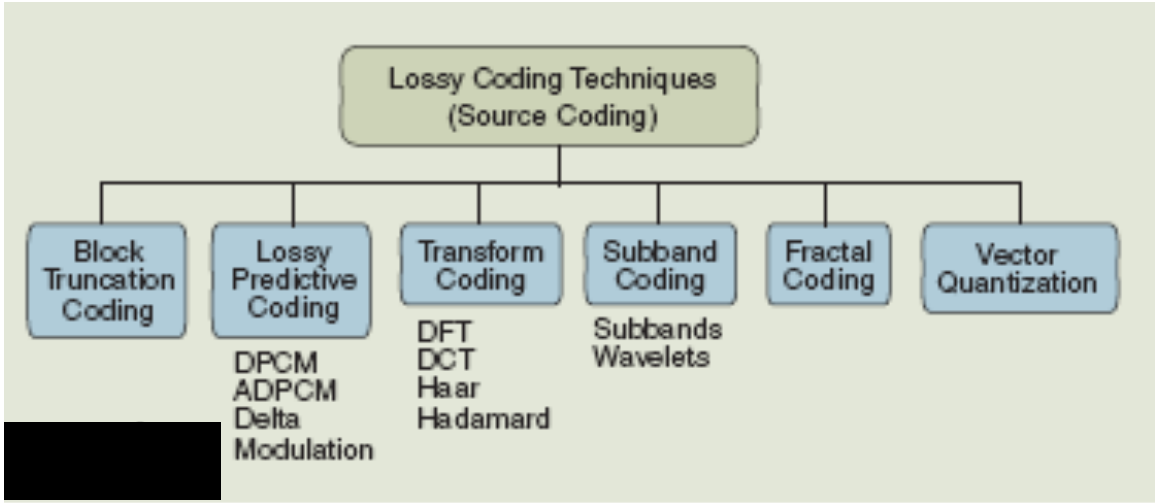
- Process each bit plane individually.

(1) Decompose an image into a series of binary images.

(2) Compress each binary image (e.g., using run-length coding)

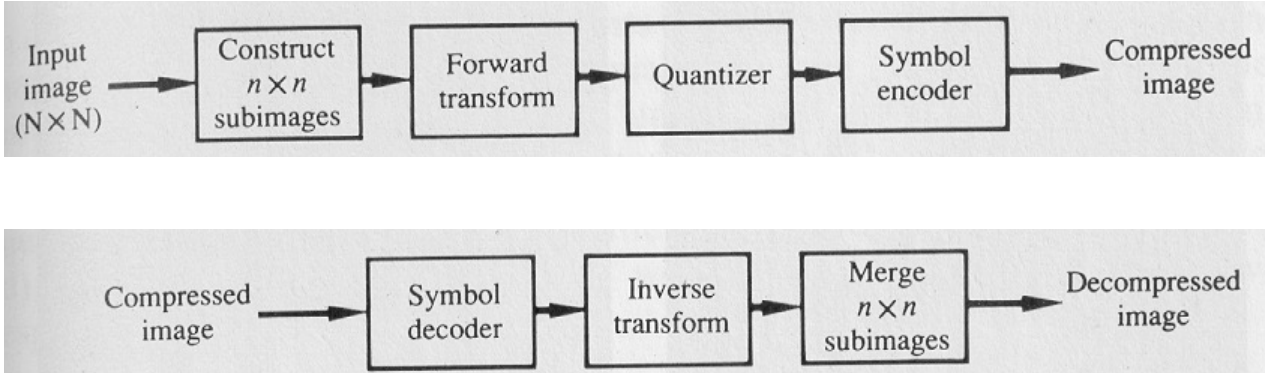


Lossy Methods - Taxonomy



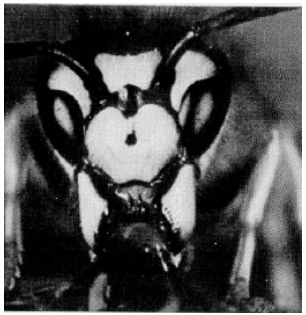
Lossy Compression

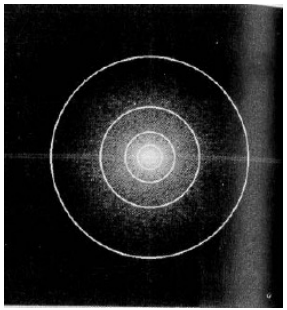
- Transform the image into some other domain to reduce interpixel redundancy.



Example: Fourier Transform

$$f(x,y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u,v) e^{\frac{j2\pi(ux+vy)}{N}}, \quad x,y=0,1,...,N-1$$





Note that the magnitude of the FT decreases, as u, v increase!

K << N

$$\hat{f}(x,y) = \frac{1}{N} \sum_{u=0}^{K-1} \sum_{v=0}^{K-1} F(u,v) e^{\frac{j2\pi(ux+vy)}{N}}, \quad x,y=0,1,...,N-1$$

$\sum_{x,y} (\hat{f}(x,y) - f(x,y))^2$ is very small !!

Transform Selection

$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u, v) h(x, y, u, v)$$

- T(u,v) can be computed using various transformations, for example:
  - DFT
  - DCT (Discrete Cosine Transform)
  - KLT (Karhunen-Loeve Transformation) or Principal Component Analysis (PCA)
  - DWT (Discrete Wavelet Transform)

- JPEG using DCT for handling interpixel redundancy.

DCT (Discrete Cosine Transform)

[proposed by Nasir Ahmed, T. Natarajan, K.R.Rao (1972)]

Forward: $$C(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left(\frac{(2x+1)u\pi}{2N}\right)\cos\left(\frac{(2y+1)v\pi}{2N}\right),$$

$u, v = 0,1,\ldots,N-1$

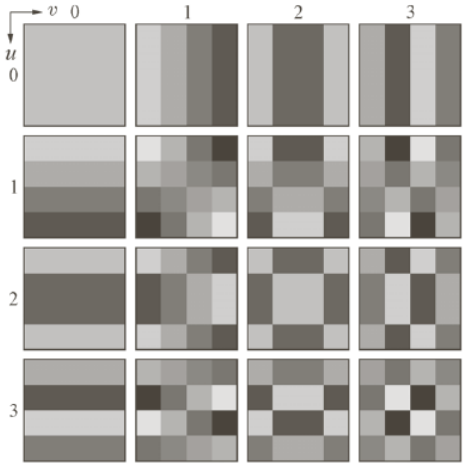
Inverse: $$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} \alpha(u)\alpha(v)C(u,v)\cos\left(\frac{(2x+1)u\pi}{2N}\right)\cos\left(\frac{(2y+1)v\pi}{2N}\right),$$

$x, y = 0,1,\ldots,N-1$

$$\alpha(u) = \begin{cases} \sqrt{1/N} & \text{if u=0} \\ \sqrt{2/N} & \text{if u>0} \end{cases} \qquad \alpha(v) = \begin{cases} \sqrt{1/N} & \text{if v=0} \\ \sqrt{2/N} & \text{if v>0} \end{cases}$$

DCT (cont'd)

- Basis functions for a 4x4 image (i.e., cosines of different frequencies).

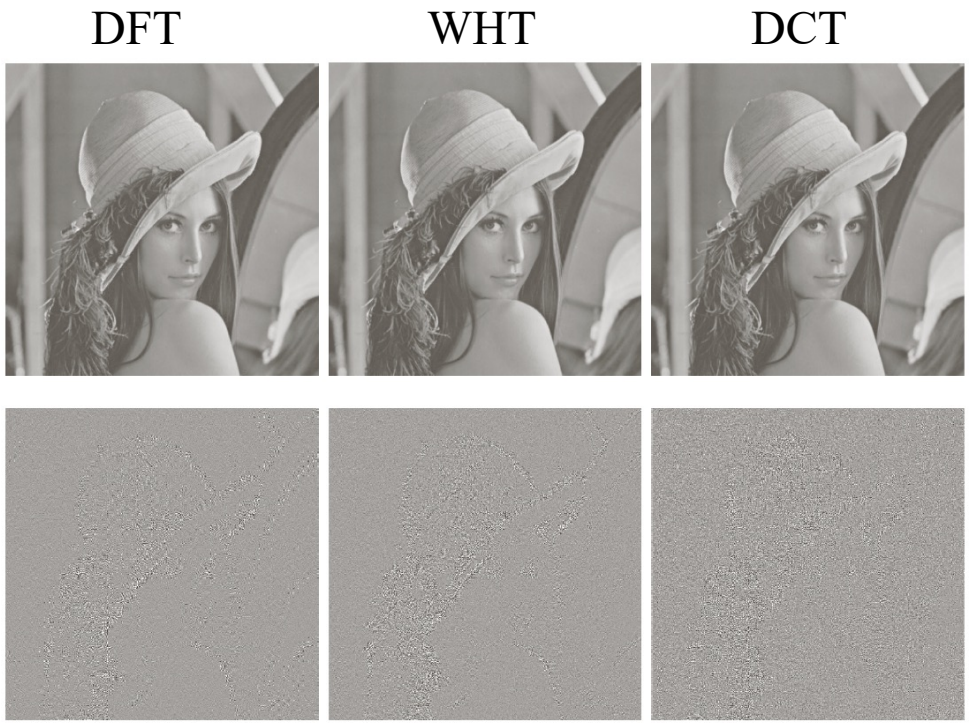


DCT (cont'd)

Using
8 x 8 sub-images
yields 64 coefficients
per sub-image.

Reconstructed images
by truncating
50% of the
coefficients

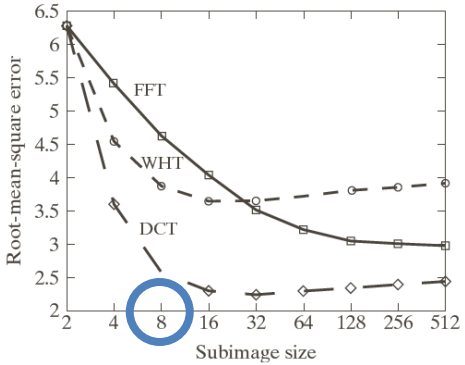
More compact
transformation



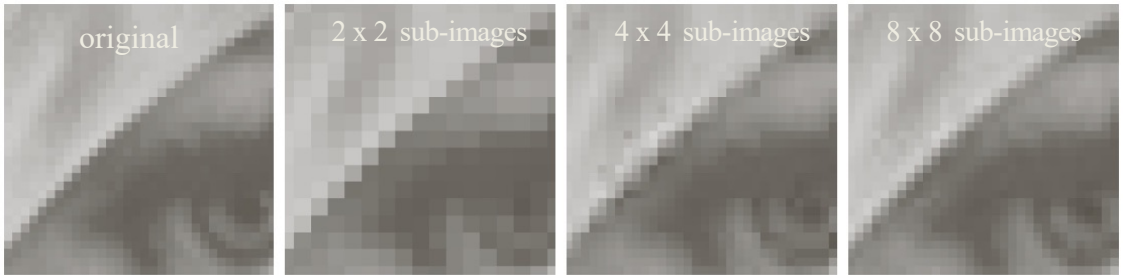
RMS error: 2.32 1.78 1.13

DCT (cont'd)

- Sub-image size selectio



Reconstructions

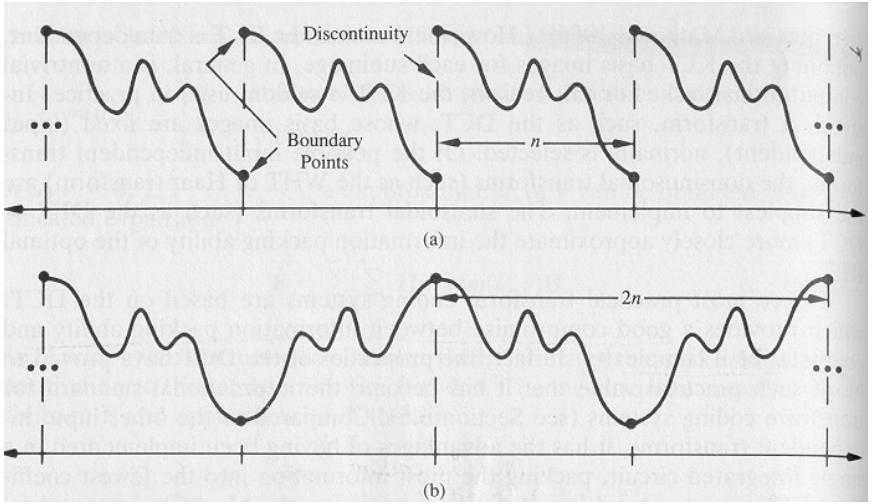


DCT (cont'd)

- DCT minimizes "blocking artifacts" (i.e., boundaries between subimages do not become very visible).

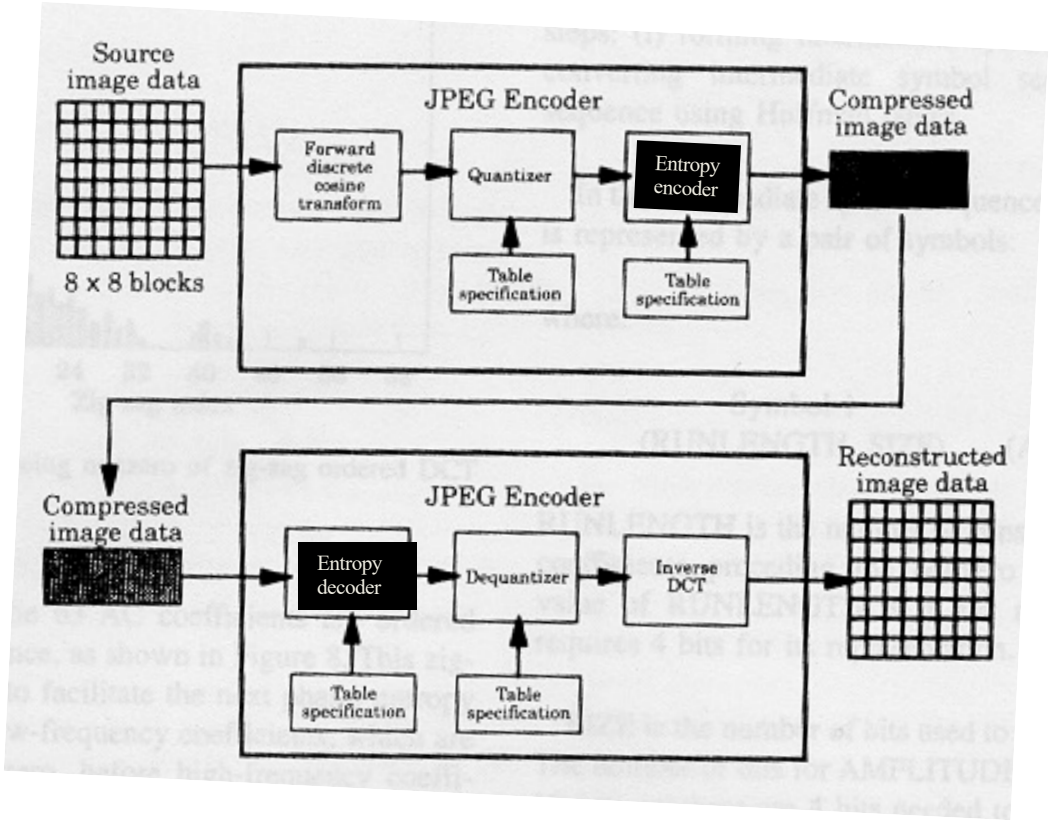
DFT
has n-point periodicity

DCT
has 2n-point periodicity



JPEG Compression

Accepted as an international image compression standard in 1992.



JPEG - Steps

1. Divide image into 8x8 subimages.

For each subimage do:

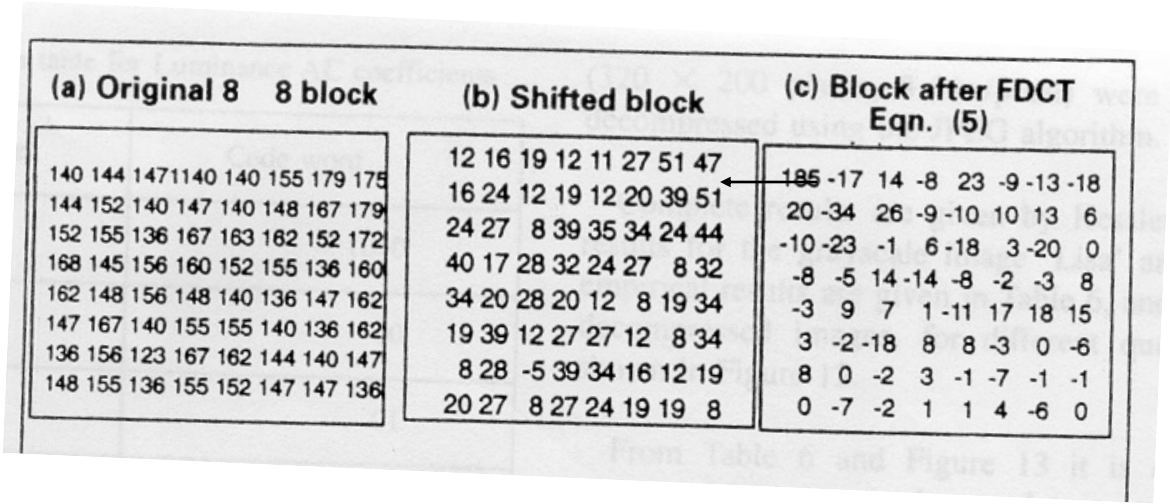
2. Shift the gray-levels in the range [-128, 127]

3. Apply DCT → 64 coefficients

   1 DC coefficient: F(0,0)

   63 AC coefficients: F(u,v)

Example



(a) Original 8 8 block

140	144	1471140	140	155	179	175	
144	152	140	147	140	148	167	179
152	155	136	167	163	162	152	172
168	145	156	160	152	155	136	160
162	148	156	148	140	136	147	162
147	167	140	155	155	140	136	162
136	156	123	167	162	144	140	147
148	155	136	155	152	147	147	136

(b) Shifted block

12	16	19	12	11	27	51	47
16	24	12	19	12	20	39	51
24	27	8	39	35	34	24	44
40	17	28	32	24	27	8	32
34	20	28	20	12	8	19	34
19	39	12	27	27	12	8	34
8	28	-5	39	34	16	12	19
20	27	8	27	24	19	19	8

(c) Block after FDCT Eqn. (5)

185	-17	14	-8	23	-9	-13	-18
20	-34	26	-9	-10	10	13	6
-10	-23	-1	6	-18	3	-20	0
-8	-5	14	-14	-8	-2	-3	8
-3	9	7	1	-11	17	18	15
3	-2	-18	8	8	-3	0	-6
8	0	-2	3	-1	-7	-1	-1
0	-7	-2	1	1	4	-6	0

[-128, 127]

(non-centered spectrum)

JPEG Steps

4. Quantize the coefficients (i.e., reduce the amplitude of coefficients that do not contribute a lot).

$$C_q(u,v) = Round[\frac{C(u,v)}{Q(u,v)}]$$

Q(u,v): quantization table

Example

- Quantization Table Q[i][j]

(d) Quantization table (quality = 2)

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31

```
for i=0 to n;
  for j=0 to n;
    Q[i,j]= 1 + (1+i+j)*quality;
  end j;
end i;
```

$1 \leq quality \leq 25$

(best - low compression)

(worst - high compression)

Example (cont'd)

(c) Block after FDCT Eqn. (5)

185	-17	14	-8	23	-9	-13	-18
20	-34	26	-9	-10	10	13	6
-10	-23	-1	6	-18	3	-20	0
-8	-5	14	-14	-8	-2	-3	8
-3	9	7	1	-11	17	18	15
3	-2	-18	8	8	-3	0	-6
8	0	-2	3	-1	-7	-1	-1
0	-7	-2	1	1	4	-6	0

(d) Quantization table (quality = 2)

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31

Quantization

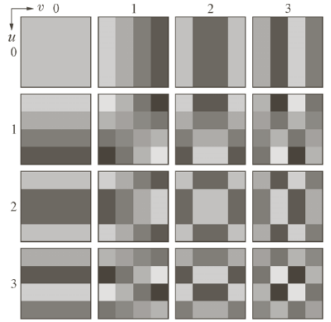
(e) Block after quantization Eqn. (6)

61	-3	2	0	2	0	0	-1
4	-4	2	0	0	0	0	0
-1	-2	0	0	-1	0	-1	0
0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	-1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

JPEG Steps (cont'd)

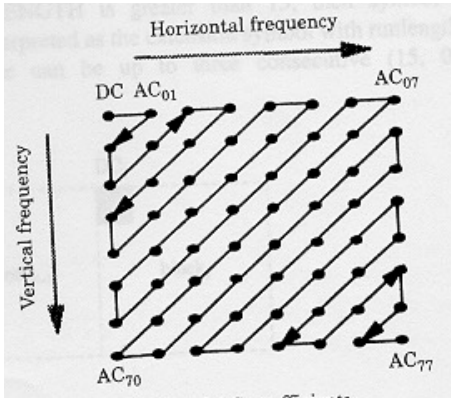
5. Order the coefficients using **zig-zag** ordering

- Creates long runs of zeros (i.e., ideal for run-length encoding)



(e) Block after quantization Eqn. (6)

61	-3	2	0	2	0	0	-1
4	-4	2	0	0	0	0	0
-1	-2	0	0	-1	0	-1	0
0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	-1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0



(f) Zig-zag sequence

61,-3,4,-1,-4,2,0,2,-2,0,0,0,0,0,2,0,0,0,1,0,0,0,0,0,0,-1,0,0,-1,0,0,
0,0,-1,0,0,0,0,0,0,0,-1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

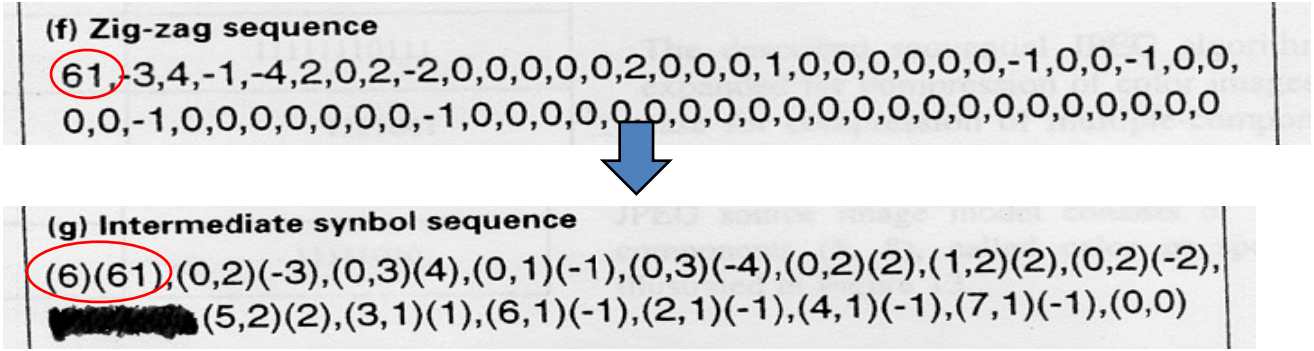
JPEG Steps (cont'd)

6. Encode coefficients:

6.1 Form "intermediate" symbol sequence.

6.2 Encode "intermediate" symbol sequence into a binary sequence.

Intermediate Symbol Sequence – DC coeff



	symbol_1 (SIZE)	symbol_2 (AMPLITUDE)
DC	(6)	(61)

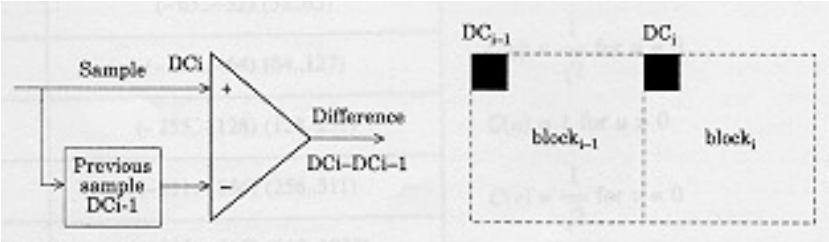
SIZE: # bits need to encode the coefficient

DC Coefficient Encoding

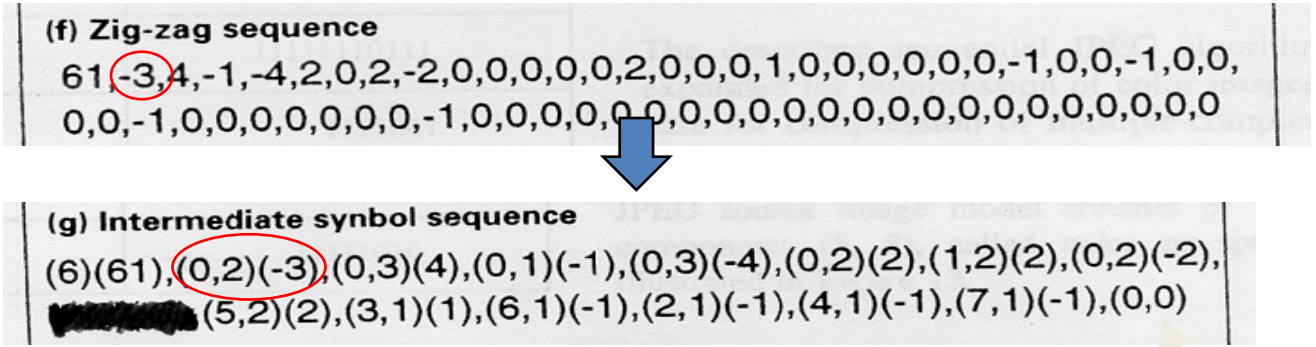
symbol_1 (SIZE)

symbol_2 (AMPLITUDE)

predictive coding:



Intermediate Symbol Sequence – AC coeff



symbol_1 (RUN-LENGTH, SIZE) symbol_2 (AMPLITUDE) end of block

AC (0, 2) (-3)

RUN-LENGTH: run of zeros preceding coefficient
SIZE: # bits for encoding the amplitude of coefficient

Note: If RUN-LENGTH > 15, use symbol (15,0) ,

Example: AC Coefficients Encoding

Symbol_1
(Variable Length Code (VLC))

(Runlength, size)	Code word
(0,,0) EOB	1010
(0,1)	00
(0,2)	01
(0,3)	100
(1,2)	11011
(2,1)	11100
(3,1)	111010
(4,1)	111011
(5,2)	11111110111
(6,1)	1111011
(7,1)	11111010

Symbol_2
(Variable Length Integer (VLI))

# bits	Amplitude range
1	(–1, 1)
2	(–3, –2) (2,3)
3	(–7..–4) (4..7)
4	(–15..–8) (8..15)
5	(–31..–16) (16..31)
6	(–63..–32) (32..63)
7	(–127..–64) (64..127)
8	(–255..–128) (128..255)
9	(–511..–256) (256..511)
10	(–1023..–512) (512..1023)

(1,4) (12) → (111110110 1100)

VLC VLI

Final Symbol Sequence

(g) Intermediate synbol sequence

(6)(61),(0,2)(-3),(0,3)(4),(0,1)(-1),(0,3)(-4),(0,2)(2),(1,2)(2),(0,2)(-2),
[redacted](5,2)(2),(3,1)(1),(6,1)(-1),(2,1)(-1),(4,1)(-1),(7,1)(-1),(0,0)

(e) Encoded bit sequence (total 98 bits)

1110111101001001000001000110110110111001011111111011
1101110101111101101110001110110111101001010

What is the effect of the “Quality” parameter?

(58k bytes)

(21k bytes)

(8k bytes)

lower compression

higher compression

$$1 \leq quality \leq 25$$


ONE DOES NOT SIMPLY
USE HIGH LEVEL OF JPEG COMPRESSION

Case Study: Fingerprint Compression

- FBI is digitizing fingerprints at 500 dots per inch with 8 bits of grayscale resolution.
- A single fingerprint card turns into about 10 MB of data!

A sample fingerprint image
768 x 768 pixels =589,824 bytes

WSQ Fingerprint Compression

- An image coding standard for digitized fingerprints employing the Discrete Wavelet Transform (*Wavelet/Scalar Quantization or WSQ)*.

- Developed and maintained by:
  - FBI
  - Los Alamos National Lab (LANL)
  - National Institute for Standards and Technology (NIST)

Need to Preserve Fingerprint Details

The "white" spots in the middle of the black ridges are *sweat pores* and they are admissible points of identification in court.

These details are just a couple pixels wide!

What compression scheme should be used?

- Lossless or lossy compression?
- In practice lossless compression methods haven’t done better than 2:1 on fingerprints!
- Does JPEG work well for fingerprint compression?

Results using JPEG compression

file size 45853 bytes
compression ratio: 12.9

Fine details have been lost.

Image has an artificial "blocky" pattern superimposed on it.

Artifacts will affect the performance of fingerprint recognition.

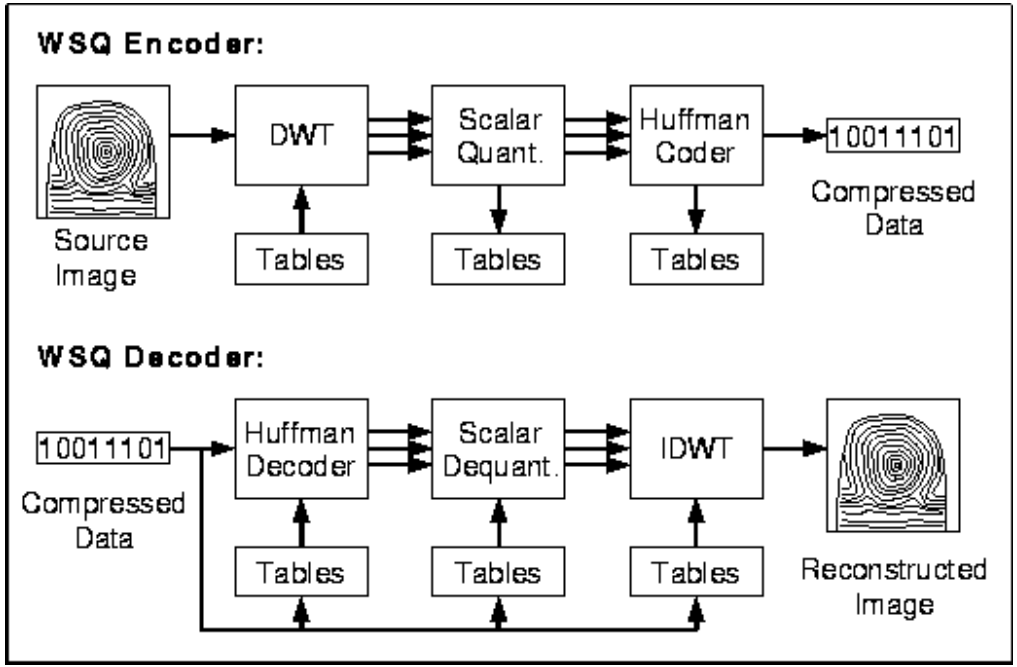
Results using WSQ compression

file size 45621 bytes
compression ratio: 12.9

Fine details are better preserved.

No “blocky” artifacts.

WSQ Algorithm



Compression ratio

- FBI’s target bit rate is around 0.75 bits per pixel (bpp)
- This corresponds to a compression ratio of 8/0.75=10.7
- Target bit rate can set via a parameter, similar to the "quality" parameter in JPEG.

Varying compression ratio (cont’d)

Original image 768 x 768 pixels (589824 bytes)

Varying compression ratio (cont'd)
0.9 bpp compression

WSQ image, file size 47619 bytes,
compression ratio 12.4

JPEG image, file size 49658 bytes,
compression ratio 11.9

Varying compression ratio (cont'd)
0.75 bpp compression

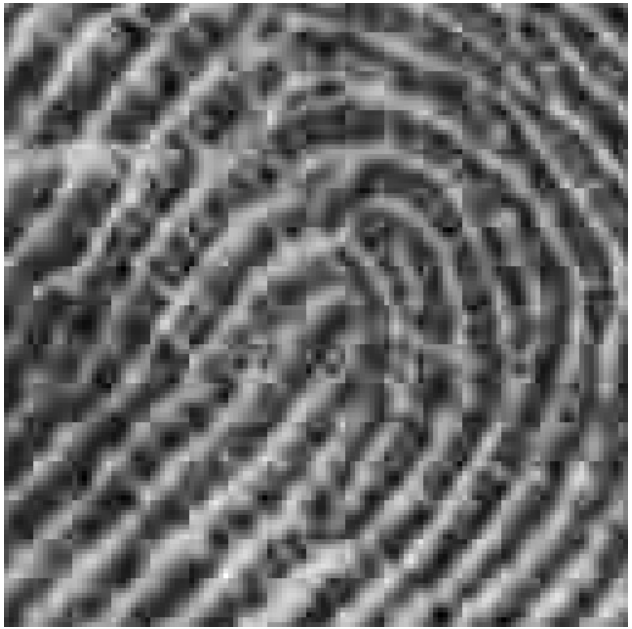
WSQ image, file size 39270 bytes
compression ratio 15.0

JPEG image, file size 40780 bytes,
compression ratio 14.5

Varying compression ratio (cont'd)
0.6 bpp compression

WSQ image, file size 30987 bytes,
compression ratio 19.0

JPEG image, file size 30081 bytes,
compression ratio 19.6



JPEG Modes

- JPEG supports several different modes
  - Sequential Mode
  - Progressive Mode
  - Hierarchical Mode
  - Lossless Mode

- The default mode is “sequential”
  - Image is encoded in a single scan (left-to-right, top-to-bottom).

Progressive JPEG

- Image is encoded in multiple scans, in order to produce a quick, rough decoded image when transmission time is long.

Sequential

Progressive

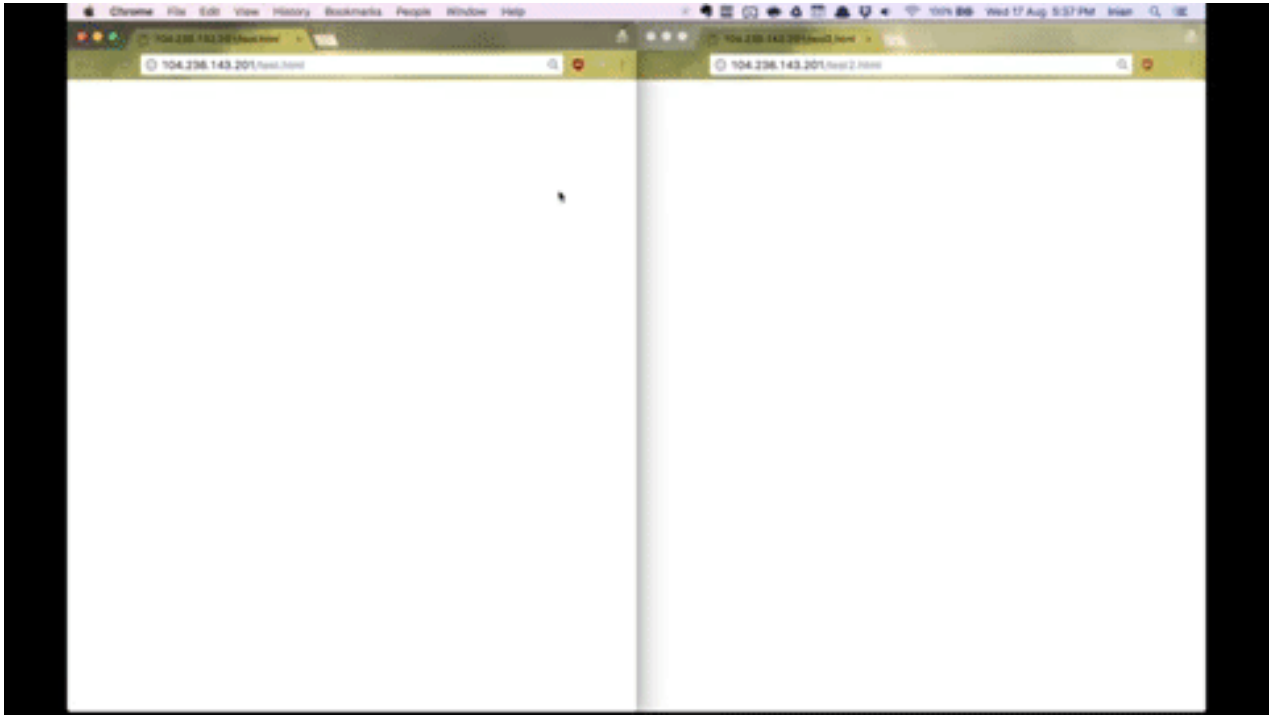
Color Images

Y

Cb

Cr

- Different quantization matrices for chrominance and luminance
- Chroma subsampling (use reduced resolution of chroma channels)


Chrome File Edit View History Bookmarks People Window Help
104.236.143.201/test.html
104.236.143.201/test2.html

JPEG at 0.125 bpp (enlarged)

C. Christopoulos, A. Skodras, T. Ebrahimi, JPEG2000 (online tutorial)

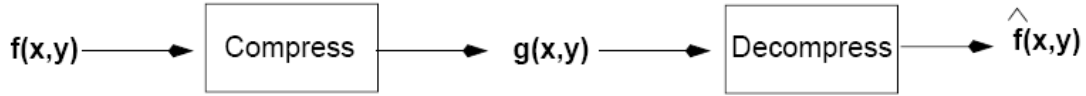
JPEG2000 at 0.125 bpp

C. Christopoulos, A. Skodras, T. Ebrahimi, JPEG2000 (online tutorial)

Other popular formats

- GIF → lossy
- PNG → lossless
- Video
  - MPEG etc. (exploit temporal redundancy also)

Fidelity Criteria



- How close is $f(x,y)$ to $\hat{f}(x,y)$?

- Criteria
  - Subjective: based on human observers
  - Objective: mathematically defined criteria

Subjective Fidelity Criteria

Value	Rating	Description
1	Excellent	An image of extremely high quality, as good as you could desire.
2	Fine	An image of high quality, providing enjoyable viewing. Interference is not objectionable.
3	Passable	An image of acceptable quality. Interference is not objectionable.
4	Marginal	An image of poor quality; you wish you could improve it. Interference is somewhat objectionable.
5	Inferior	A very poor image, but you could watch it. Objectionable interference is definitely present.
6	Unusable	An image so bad that you could not watch it.

Quality measurement: judged by human viewers

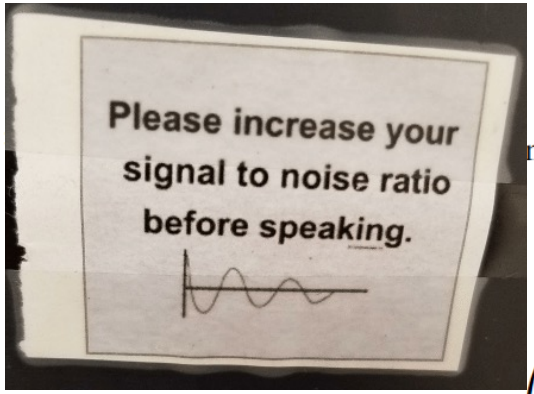
- Five scale system on the degree of impairment
  1. Impairment is not noticeable
  2. Impairment is just noticeable
  3. Impairment is definitely noticeable, but not objectionable
  4. Impairment is objectionable
  5. Impairment is extremely objectionable

Advantages: relies on HVS

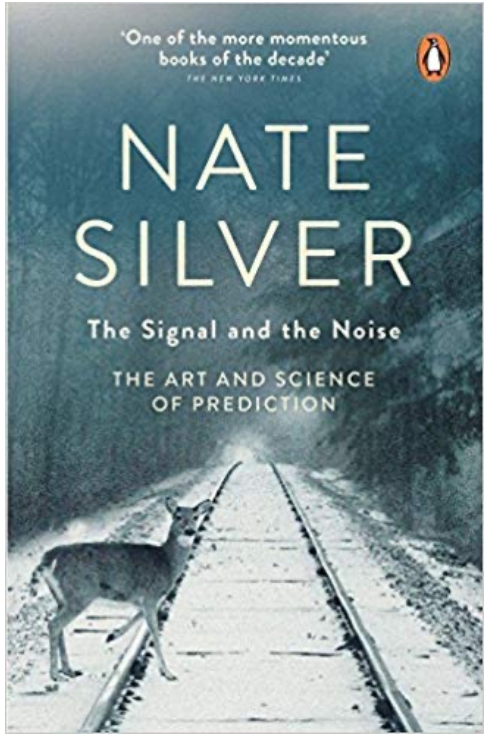
Drawbacks: time, viewing conditions, viewers?

Quality measurement: Signal to noise ratio

$e(x, y)$



$$\text{ms} = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} e(x, y)^2$$



1

$$SNR_{ms} = 10 \log_{10} \left(\frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} g(x, y)^2}{MN \cdot E_{\text{ms}}} \right)$$

Decibel (dB)

Quality measurement: Signal to noise ratio

$$e(x,y) = f(x,y) - g(x,y).$$

$$E_{\text{ms}} = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} e(x,y)^2$$

1 $$SNR_{ms} = 10 \log_{10} \left(\frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} g(x,y)^2}{MN \cdot E_{\text{ms}}} \right)$$

Decibel (dB)

Peak SNR

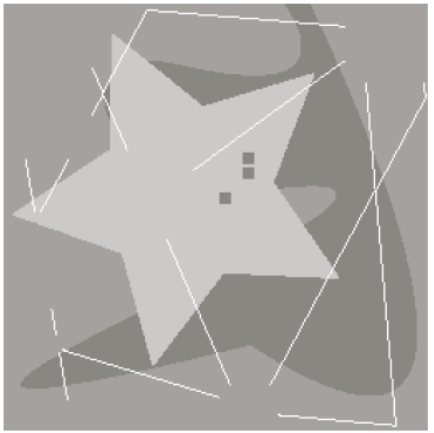
2 $$PSNR = 10 \log_{10} \left(\frac{255^2}{E_{\text{ms}}} \right)$$

Subjective vs Objective Fidelity Criteria

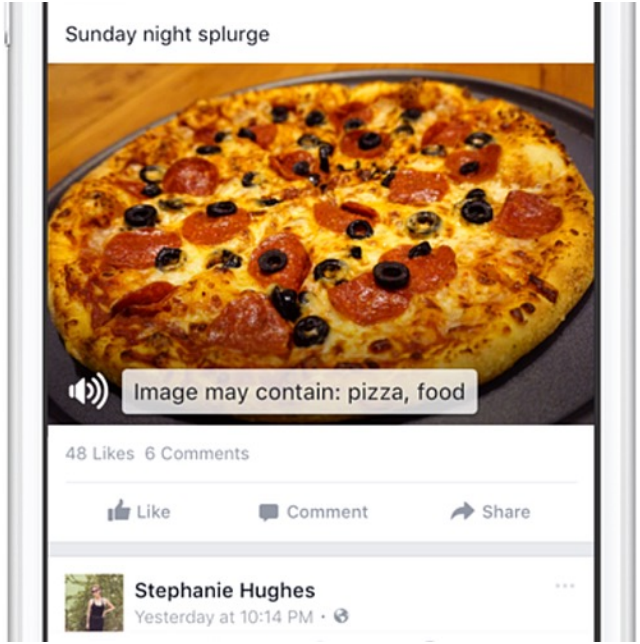
RMSE = 5.17

RMSE = 15.67

RMSE = 14.17



Next generation tech: ‘Semantic’ data compression !



http://www.knowahead.in/wp-content/uploads/2017/03/fb_image-1.png

http://www.networkedindia.com/wp-content/uploads/2016/04/facebook-iOS-app-blind-ai.png

Humans are still the best lossy image compressors

Ashutosh Bhown[1,*], Soham Mukherjee[2,*], Sean Yang[3,*],
Shubham Chandak[4], Irena Fischer-Hwang[4], Kedar Tatwawadi[4], Tsachy Weissman[4]

[1]Palo Alto High School [2]Monta Vista High School
[3]Saint Francis High School [4]Stanford University
schandak@stanford.edu

Abstract

Lossy image compression has been studied extensively in the context of typical loss functions such as RMSE, MS-SSIM, etc. However, it is not well understood what loss function might be most appropriate for human perception. Furthermore, the availability of massive public image datasets appears to have hardly been exploited in image compression. In this work, we perform compression experiments in which one human describes images to another, using publicly available images and text instructions. These image reconstructions are rated by human scorers on the Amazon Mechanical Turk platform and compared to reconstructions obtained by existing image compressors. In our experiments, the humans outperform the state of the art compressor WebP in the MTurk survey on most images, which shows that there is significant room for improvement in image compression for human perception.
Data: The images, results and additional data is available at `https://compression.stanford.edu/human-compression`.

Introduction

Since the advent of electronic media, image compression has been studied extensively, leading to multiple image formats and compression techniques such as PNG [1], JPEG [2], JPEG2000 [3], JPEG XR [4], BPG [5] and WebP [6]. In order to

(a) Original (b) WebP (c) Human reconstruction

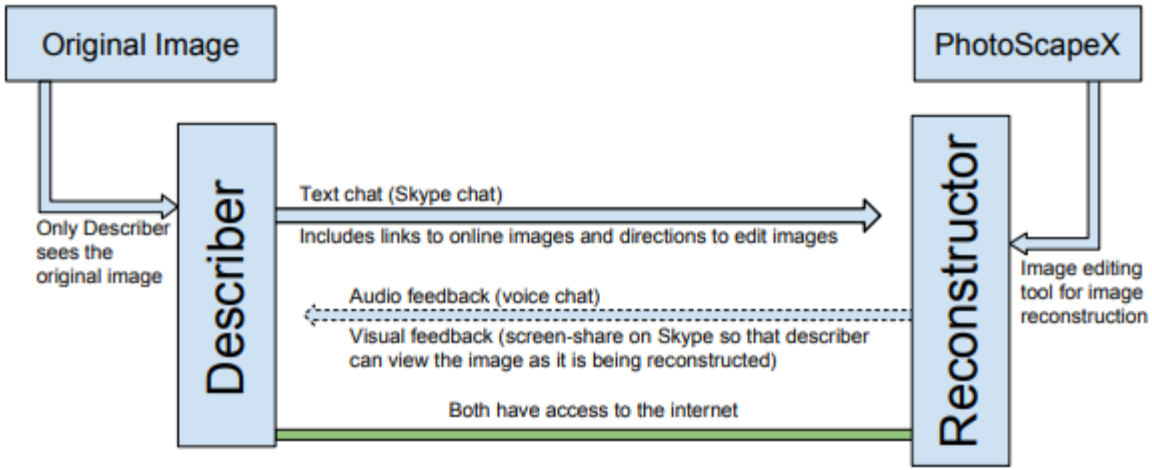


Figure 2: Block diagram showing the human compression process. The describer attempts to describe the image with URL links and text instructions. The describer can see the image as it is being reconstructed and can also hear the reconstructor's voice.

https://www.worldwildlife.org/habitats/grasslands

try transformations
elongate the fence bit
only focus on the vertical…

there's a line of shrubbery that goes across the middle third of the image…
that's the largest bush in the pic
so keep the others sizes equal to or smaller that that and make it look continuous
and make sure to make the bushes smaller as you work your way up so that there's a sense of depth…

there will be a line of tiny shrubs along that line…
the line itself starts about a quarter from the left…

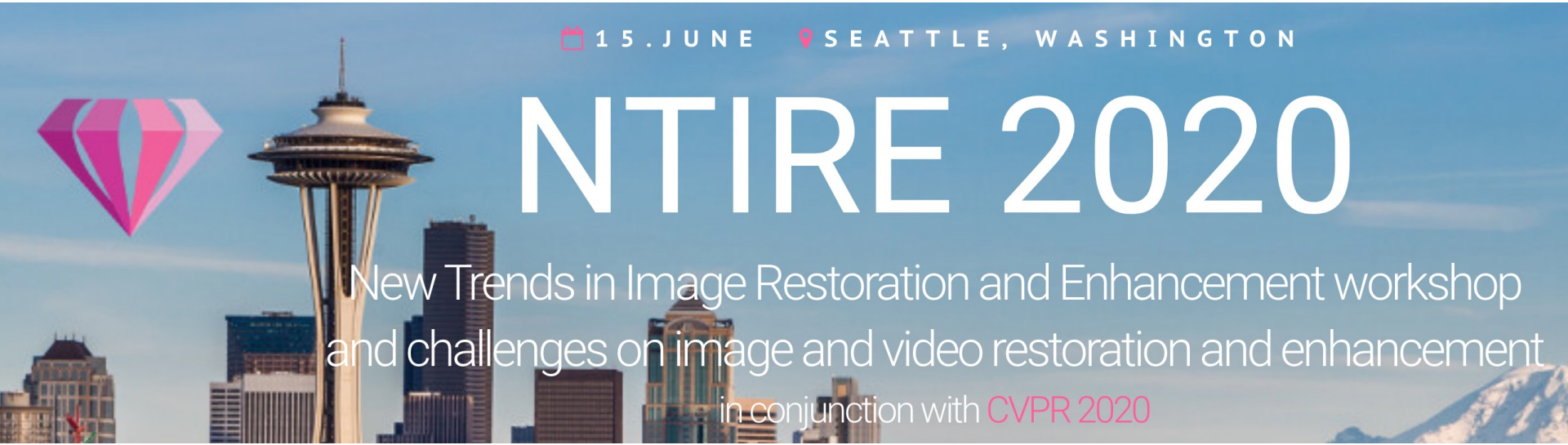
Try and make the grass look less tall on the bottom…

when you're done with that take a look at these
https://public-media.smithsonianmag.com/filer/32/f2/32f24473-b380-43f5-94df-da0e58644439/16301090250_acf80be87f_o.jpg
https://img.purch.com/w/192/aHR0cDovL3d3dy5saXZlc2NpZW5jZS5jb20vaW1hZ2VzL2kvMDAwLzA2OC8wOTQvaTMwMC9naXJhZmZlLmpwZz8xNDA1MDA4NDQy
sure
while you're editing that giraffe
its spots are too dark
make it look like the other giragge…

make the right one bigger than the left
make the heads level
wait back
put the left one where it was before
good
now move the right giraffe to the left so that their necks cross
good
move them both to the center
make them both taller as well
their heads should be above the middle line of shrubs…

Final reconstruction

Original image


15.JUNE
SEATTLE, WASHINGTON
NTIRE 2020
New Trends in Image Restoration and Enhancement workshop
and challenges on image and video restoration and enhancement
in conjunction with CVPR 2020

Reference

- Ch 8, G&W textbook