

# Image and Video Compression

Lecture 12, April 28<sup>th</sup>, 2008

Lexing Xie

EE4830 Digital Image Processing

<http://www.ee.columbia.edu/~xix/ee4830/>

material sources: David McKay's book, Min Wu (UMD), Yao Wang (poly tech), ...

# Announcements

# Lecture Outline

- Image/Video compression: What and why
- Source coding
  - basic idea
  - symbol codes
  - stream codes
- Compression systems and standards
  - JPEG
  - MPEG
- Summary

## the need for compression

- Image: 6.0 million pixel camera, 3000x2000
  - 18 MB per image → 56 pictures / 1GB
- Video: DVD Disc 4.7 GB
  - video 720x480, RGB, 30 f/s → 31.1MB/sec
  - audio 16bits x 44.1KHz stereo → 176.4KB/s
    - → 1.5 min per DVD disc
- Send video from cellphone:  
352\*240, RGB, 15 frames / second
  - 3.8 MB/sec → \$38.00/sec levied by AT&T

# Data Compression

- Wikipedia: “data compression, or source coding, is the process of encoding information using fewer bits (or other information-bearing units) than an unencoded representation would use through use of specific encoding schemes.”
- Applications
  - General data compression: .zip, .gz ...
  - Image over network: telephone/internet/wireless/etc
  - Slow device:
    - 1xCD-ROM 150KB/s, bluetooth v1.2 up to ~0.25MB/s
  - Large multimedia databases

# what can we compress?

- Goals of compression
  - Remove **redundancy**
  - Reduce **irrelevance**
- irrelevance or perceptual redundancy
  - not all visual information is perceived by eye/brain, so throw away those that are not.

a b c  
**FIGURE 8.4**  
(a) Original image.  
(b) Uniform quantization to 16 levels. (c) IGS quantization to 16 levels.



# what can we compress?

- Goals of compression
  - Remove redundancy
  - Reduce irrelevance
- redundant : exceeding what is necessary or normal
  - symbol redundancy
    - the common and uncommon values cost the same to store
  - spatial and temporal redundancy
    - adjacent pixels are highly correlated.

# symbol/inter-symbol redundancy

- Letters and words in English
  - e, a, i, s, t, ...  
q, y, z, x, j, ...
  - a, the, me, I ...  
good, magnificent, ...
  - fyi, btw, ttyl ...
- In the evolution of language we naturally chose to represent frequent meanings with shorter representations.

## INTERNATIONAL MORSE CODE

1. A dash is equal to three dots.
2. The space between parts of the same letter is equal to one dot.
3. The space between two letters is equal to three dots.
4. The space between two words is equal to five dots.

A • —  
 B — • • •  
 C — • — •  
 D — • •  
 E •  
 F • • — •  
 G — — •  
 H • • • •  
 I • •  
 J • — — —  
 K — • —  
 L • — • •  
 M — —  
 N — •  
 O — — —  
 P • — — •  
 Q — — • —  
 R • — •  
 S • • •  
 T —

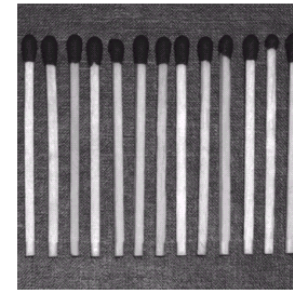
U • • —  
 V • • • —  
 W • — —  
 X • • • —  
 Y — • — —  
 Z — — • •

1 • — — — —  
 2 • • — — —  
 3 • • • — —  
 4 • • • • —  
 5 • • • • •  
 6 — • • • •  
 7 — — • • •  
 8 — — — • •  
 9 — — — — •  
 0 — — — — —

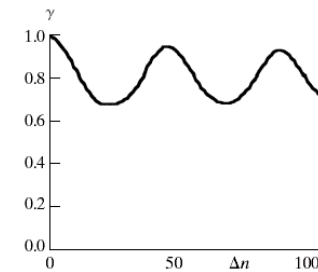
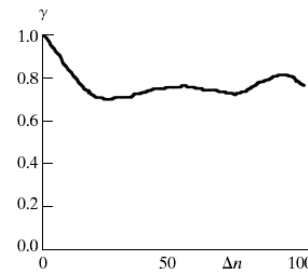
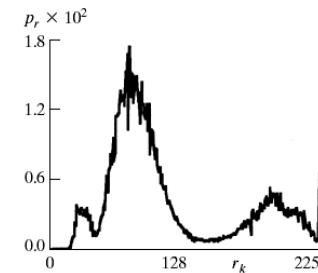
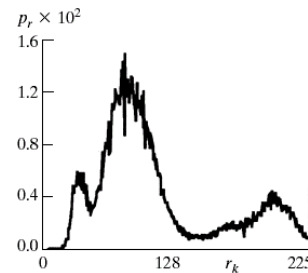


# pixel/inter-pixel redundancy

- Some gray level value are more probable than others.
- Pixel values are not i.i.d. (independent and identically distributed)



**FIGURE 8.2** Two images and their gray-level histograms and normalized autocorrelation coefficients along one line.



# modes of compression

- Lossless
  - preserve all information, perfectly recoverable
  - examples: morse code, zip/gz
- Lossy
  - cannot recover all bits



# characterizing information

## ■ i.i.d. random variable $x$

An ensemble  $X$  is a triple  $(x, \mathcal{A}_X, \mathcal{P}_X)$ , where the *outcome*  $x$  is the value of a random variable, which takes on one of a set of possible values,  $\mathcal{A}_X = \{a_1, a_2, \dots, a_i, \dots, a_I\}$ , having probabilities  $\mathcal{P}_X = \{p_1, p_2, \dots, p_I\}$ , with  $P(x = a_i) = p_i$ ,  $p_i \geq 0$  and  $\sum_{a_i \in \mathcal{A}_X} P(x = a_i) = 1$ .

## ■ information content

- characterize the surprising-ness
- related to probability
- additive for independent variables.

$$h(x = a_i) = \log_2 \frac{1}{p_i}$$

## ■ explanations

- cross-words: how many words have you "ruled out" after knowing that a word starts with an "a" or with a "z" ?

#"a\*" 35,174 words      #"z\*" 1,718 words

English vocabulary: ~500K words

$i$	$a_i$	$p_i$	$h(p_i)$
1	a	.0575	4.1
2	b	.0128	6.3
3	c	.0263	5.2
4	d	.0285	5.1
5	e	.0913	3.5
6	f	.0173	5.9
7	g	.0133	6.2
8	h	.0313	5.0
9	i	.0599	4.1
10	j	.0006	10.7
11	k	.0084	6.9
12	l	.0335	4.9
13	m	.0235	5.4
14	n	.0596	4.1
15	o	.0689	3.9
16	p	.0192	5.7
17	q	.0008	10.3
18	r	.0508	4.3
19	s	.0567	4.1
20	t	.0706	3.8
21	u	.0334	4.9
22	v	.0069	7.2
23	w	.0119	6.4
24	x	.0073	7.1
25	y	.0164	5.9
26	z	.0007	10.4
27	-	.1928	2.4
$\sum_i p_i \log_2 \frac{1}{p_i}$			4.1



# information content and entropy

## ■ Shannon information content

$$h(x = a_i) = \log_2 \frac{1}{p_i}$$

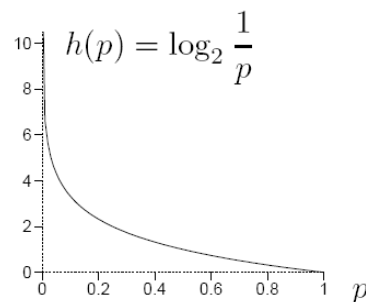
- additive for independent variables:

$$\begin{aligned} h(x = a_i, y = a_j) &= \log_2 \frac{1}{p_i p_j} \\ &= h(x = a_i) + h(y = a_j) \end{aligned}$$

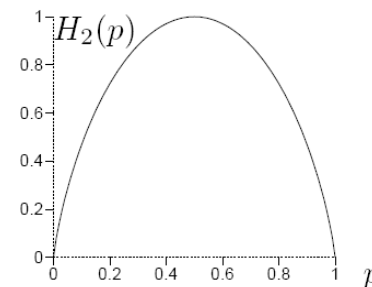
## ■ Entropy: expected information content

$$H(X) = E\{h(x)\} = \sum_{a_i \in \mathcal{A}_X} p_i \log_2 \frac{1}{p_i}$$

$i$	$a_i$	$p_i$	$h(p_i)$
1	a	.0575	4.1
2	b	.0128	6.3
3	c	.0263	5.2
4	d	.0285	5.1
5	e	.0913	3.5
6	f	.0173	5.9
7	g	.0133	6.2
8	h	.0313	5.0
9	i	.0599	4.1
10	j	.0006	10.7
11	k	.0084	6.9
12	l	.0335	4.9
13	m	.0235	5.4
14	n	.0596	4.1
15	o	.0689	3.9
16	p	.0192	5.7
17	q	.0008	10.3
18	r	.0508	4.3
19	s	.0567	4.1
20	t	.0706	3.8
21	u	.0334	4.9
22	v	.0069	7.2
23	w	.0119	6.4
24	x	.0073	7.1
25	y	.0164	5.9
26	z	.0007	10.4
27	-	.1928	2.4



$p$	$h(p)$	$H_2(p)$
0.001	10.0	0.011
0.01	6.6	0.081
0.1	3.3	0.47
0.2	2.3	0.72
0.5	1.0	1.0



$$\sum_i p_i \log_2 \frac{1}{p_i} = 4.1$$

## source coding

consider ensemble  $X : (x, \mathcal{A}_x, \mathcal{P}_x)$

- source code  $c(x) : \mathcal{A}_x \rightarrow \mathcal{C}_x$
- length of a codeword  $l(c(x)), x \in \mathcal{A}_x$

- expected length of a code

$$L(C, X) = \sum_{a_i \in \mathcal{A}_x} p_i l(c(a_i))$$

- an example

$\mathcal{A}_x = \{a, b, c, d\}$	$P(X = a) = 1/2$	$C(a) = 1$
	$P(X = b) = 1/4$	$C(b) = 10$
	$P(X = c) = 1/8$	$C(c) = 110$
	$P(X = d) = 1/8$	$C(d) = 111$

$$H(X) = ? , \quad L(C, X) = ?$$

# source coding theorem

informal shorthand:  $L(C, X) \geq H(X)$

**Shannon's source coding theorem (verbal statement).**  $N$  i.i.d. random variables each with entropy  $H(X)$  can be compressed into more than  $NH(X)$  bits with negligible risk of information loss, as  $N \rightarrow \infty$ ; conversely if they are compressed into fewer than  $NH(X)$  bits it is virtually certain that information will be lost.

**Theorem 4.1** Shannon's source coding theorem. *Let  $X$  be an ensemble with entropy  $H(X) = H$  bits. Given  $\epsilon > 0$  and  $0 < \delta < 1$ , there exists a positive integer  $N_0$  such that for  $N > N_0$ ,*

$$\left| \frac{1}{N} H_\delta(X^N) - H \right| < \epsilon. \quad (4.21)$$

[Shannon 1948]

# revisiting Morse code

## INTERNATIONAL MORSE CODE

1. A dash is equal to three dots.
2. The space between parts of the same letter is equal to one dot.
3. The space between two letters is equal to three dots.
4. The space between two words is equal to five dots.

<b>A</b> • —	<b>U</b> • • —
<b>B</b> — • • •	<b>V</b> • • • —
<b>C</b> — • — •	<b>W</b> • — —
<b>D</b> — • •	<b>X</b> — • • —
<b>E</b> •	<b>Y</b> — • — —
<b>F</b> • • — •	<b>Z</b> — — • •
<b>G</b> — — •	
<b>H</b> • • • •	
<b>I</b> • •	
<b>J</b> • — — —	
<b>K</b> — • —	<b>1</b> • — — — —
<b>L</b> • — • •	<b>2</b> • • — — —
<b>M</b> — —	<b>3</b> • • • — —
<b>N</b> — •	<b>4</b> • • • • —
<b>O</b> — — —	<b>5</b> • • • • •
<b>P</b> • — — •	<b>6</b> — • • • •
<b>Q</b> — — • —	<b>7</b> — — • • •
<b>R</b> • — •	<b>8</b> — — — • •
<b>S</b> • • •	<b>9</b> — — — — •
<b>T</b> —	<b>0</b> — — — — —

$i$	$a_i$	$p_i$	$h(p_i)$
1	a	.0575	4.1
2	b	.0128	6.3
3	c	.0263	5.2
4	d	.0285	5.1
5	e	.0913	3.5
6	f	.0173	5.9
7	g	.0133	6.2
8	h	.0313	5.0
9	i	.0599	4.1
10	j	.0006	10.7
11	k	.0084	6.9
12	l	.0335	4.9
13	m	.0235	5.4
14	n	.0596	4.1
15	o	.0689	3.9
16	p	.0192	5.7
17	q	.0008	10.3
18	r	.0508	4.3
19	s	.0567	4.1
20	t	.0706	3.8
21	u	.0334	4.9
22	v	.0069	7.2
23	w	.0119	6.4
24	x	.0073	7.1
25	y	.0164	5.9
26	z	.0007	10.4
27	-	.1928	2.4

$$\sum_i p_i \log_2 \frac{1}{p_i} \quad 4.1$$



# desired properties of symbol codes

- good codes are not only short but also easy to encode/decode
  - Non-singular: every symbol in  $X$  maps to a different code word
  - Uniquely decodable: every sequence  $\{x_1, \dots, x_n\}$  maps to different codeword sequence
  - Instantaneous: no codeword is a prefix of any other codeword

## a joke: English in less than 26 letters

The European Union commissioners have announced that agreement has been reached to adopt English as the preferred language for European communications, rather than German, which was the other possibility. As part of the negotiations, Her Majesty's Government conceded that English spelling had some room for improvement and has accepted a five-year phased plan for what will be known as Euro-English (Euro for short).

In the first year, 's' will be used instead of the soft 'c'. Certainly, sivil servants will resieve this news with joy. Also, the hard 'c' will be replaced with 'k.' Not only will this klear up konfusion, but typewriters kan have one less letter.

There will be growing publik enthusiasm in the sekond year, when the troublesome 'ph' will be replaced by 'f'. This will make words like 'fotograf' 20 per sent shorter.

In the third year, publik akseptanse of the new spelling kan be expekted to reach the stage where more komplikated changes are possible. Governments will enkourage the removal of double letters, which have always ben a deterrent to akurate speling. Also, al wil agre that the horrible mes of silent 'e's in the languag is disgrasful, and they would go.

By the fourth year, peopl wil be reseptiv to steps such as replasing 'th' by 'z' and 'W' by 'V'. During ze fifz year, ze unesesary 'o' kan be dropd from vords kontaining 'ou', and similar changes vud of kors; be aplid to ozer kombinations of leters.

After zis fifz yer, ve vil hav a reli sensibl riten styl. Zer vil b no mor trubls or difikultis and evrivun vil find it ezi tu understand ech ozer. Ze drem vil finali kum tru.

# desired properties of symbol codes

- Non-singular: every symbol in  $X$  maps to a different code word
- Uniquely decodable: every sequence  $\{x_1, \dots, x_n\}$  maps to different codeword sequence
- Instantaneous: no codeword is a prefix of any other codeword

## INTERNATIONAL MORSE CODE

1. A dash is equal to three dots.
2. The space between parts of the same letter is equal to one dot.
3. The space between two letters is equal to three dots.
4. The space between two words is equal to five dots.

A	• —	U	• • —
B	— • • •	V	• • • —
C	— • — •	W	• — —
D	— • •	X	— • • —
E	•	Y	— • — —
F	• • — •	Z	— — • •
G	— — •		
H	• • • •		
I	• •		
J	• — — —		
K	— • —		
L	• — • •		
M	— —		
N	— •		
O	— — —		
P	• — — •		
Q	— — • —		
R	• • •		
S	• • •		
T	—		

1	• — — — —
2	• • — — —
3	• • • — —
4	• • • • —
5	• • • • •
6	— • • • •
7	— — • • •
8	— — — • •
9	— — — — •
0	— — — — —

Morse code without blanks:

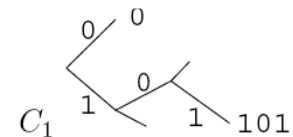
EAH      ←      • • — • • • •

IDI      ←

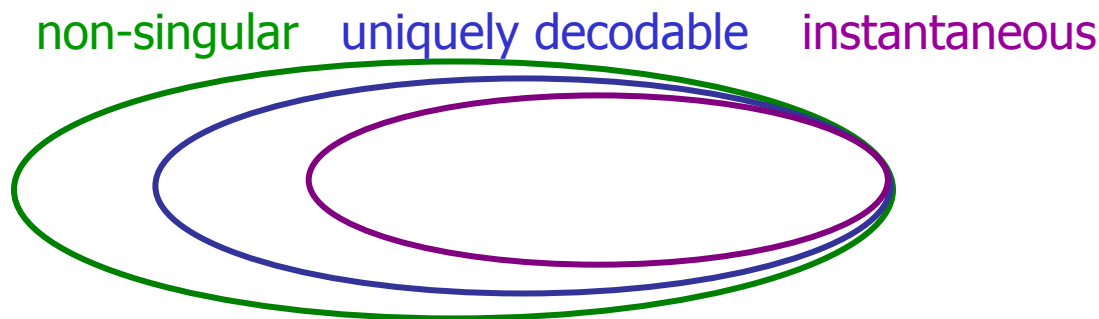
# desired properties of symbol codes

- Non-singular: every symbol in  $X$  maps to a different code word
- Uniquely decodable: every sequence  $\{x_1, \dots, x_n\}$  maps to different codeword sequence
- Instantaneous: no codeword is a prefix of any other codeword  
a.k.a prefix code, self-punctuating code, prefix-free code.

Example 5.4. The code  $C_1 = \{0, 101\}$  is a prefix code because 0 is not a prefix of 101, nor is 101 a prefix of 0.



Example 5.5. Let  $C_2 = \{1, 101\}$ . This code is not a prefix code because 1 is a prefix of 101.



good news: being unique decodable + instantaneous do not compromise coding efficiency (much)

The optimal symbol code's expected length  $L$  satisfies

$$H(X) \leq L < H(X) + 1$$

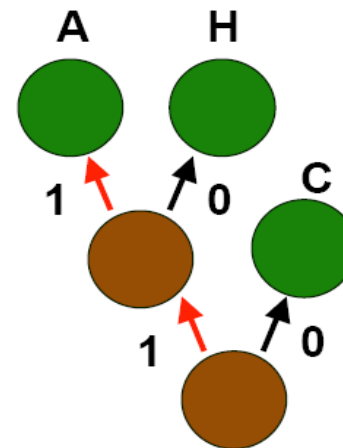
# Huffman codes

- optimal symbol code by construction

- **Binary (Huffman) tree**

- Represents Huffman code
- Edge  $\Rightarrow$  code (0 or 1)
- Leaf  $\Rightarrow$  symbol
- Path to leaf  $\Rightarrow$  encoding
- Example

- A = "11", H = "10", C = "0"



- **Encoding**

1. Calculate frequency of symbols in file
2. Create binary tree representing "best" encoding
3. Use binary tree to encode compressed file
  - For each symbol, output path from root to leaf
  - Size of encoding = length of path
4. Save binary tree

# construct Huffman codes

## ■ a recursive algorithm in two steps

1. Take the two least probable symbols in the alphabet. These two symbols will be given the longest codewords, which will have equal length, and differ only in the last digit.
2. Combine these two symbols into a single symbol, and repeat.

## ■ an example

Example 5.18. Find the optimal binary symbol code for the ensemble:

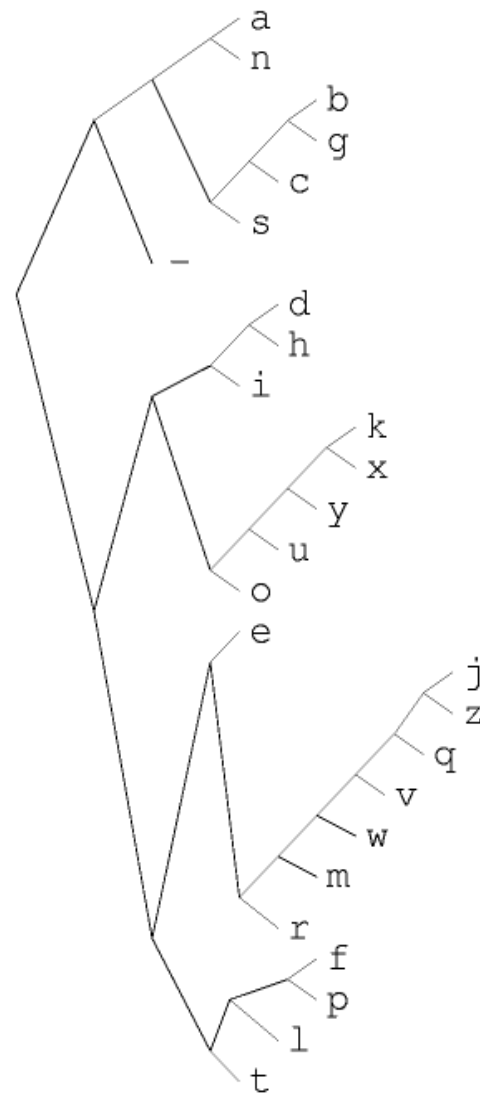
$$\mathcal{A}_X = \{ a, b, c, d, e, f, g \}$$

$$\mathcal{P}_X = \{ 0.01, 0.24, 0.05, 0.20, 0.47, 0.01, 0.02 \} .$$

$a_i$	$p_i$	Greedy	Huffman
a	.01	000	000000
b	.24	001	01
c	.05	010	0001
d	.20	011	001
e	.47	10	1
f	.01	110	000001
g	.02	111	00001

Table 5.7. A greedily-constructed code compared with the Huffman code.

$a_i$	$p_i$	$\log_2 \frac{1}{p_i}$	$l_i$	$c(a_i)$
a	0.0575	4.1	4	0000
b	0.0128	6.3	6	001000
c	0.0263	5.2	5	00101
d	0.0285	5.1	5	10000
e	0.0913	3.5	4	1100
f	0.0173	5.9	6	111000
g	0.0133	6.2	6	001001
h	0.0313	5.0	5	10001
i	0.0599	4.1	4	1001
j	0.0006	10.7	10	1101000000
k	0.0084	6.9	7	1010000
l	0.0335	4.9	5	11101
m	0.0235	5.4	6	110101
n	0.0596	4.1	4	0001
o	0.0689	3.9	4	1011
p	0.0192	5.7	6	111001
q	0.0008	10.3	9	110100001
r	0.0508	4.3	5	11011
s	0.0567	4.1	4	0011
t	0.0706	3.8	4	1111
u	0.0334	4.9	5	10101
v	0.0069	7.2	8	11010001
w	0.0119	6.4	7	1101001
x	0.0073	7.1	7	1010001
y	0.0164	5.9	6	101001
z	0.0007	10.4	10	1101000001
-	0.1928	2.4	2	01



# why do we need stream codes

- Huffman code is optimal but must be integer length.
- the interval  $[H(X), H(X)+1)$  can be loose.
- consider the following optimal symbol code:

a	0.001	00000
b	0.001	00001
c	0.990	1
d	0.001	00010
e	0.001	00011
f	0.001	0100
g	0.001	0101
h	0.001	0110
i	0.001	0111
j	0.001	0010
k	0.001	0011

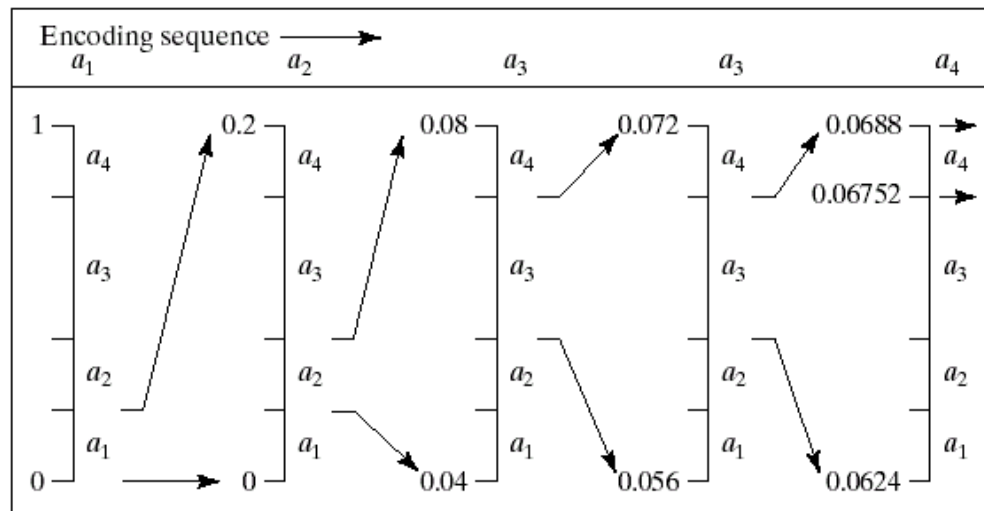
expected length	1.034
entropy	0.11401
length / entropy	9



# arithmetic coding

Source Symbol	Probability	Initial Subinterval
$a_1$	0.2	$[0.0, 0.2)$
$a_2$	0.2	$[0.2, 0.4)$
$a_3$	0.4	$[0.4, 0.8)$
$a_4$	0.2	$[0.8, 1.0)$

**TABLE 8.6**  
Arithmetic coding  
example.



**FIGURE 8.13**  
Arithmetic coding  
procedure.

# universal data compression

- What if the symbol probabilities are unknown?
- LZW algorithm (Lempel-Ziv-Welch)

## encoding

```
w = NIL;
while ( read a character k )
{
    if wk exists in the dictionary
        w = wk;
    else
        add wk to the dictionary;
        output the code for w;
        w = k;
}
```

## decoding

```
read a character k;
output k;
w = k;
while ( read a character k )
/* k could be a character or a code. */
{
    entry = dictionary entry for k;
    output entry;
    add w + entry[0] to dictionary;
    w = entry;
}
```

- Widely used: GIF, TIFF, PDF ...
- Its royalty-free variant (DEFLATE) used in PNG, ZIP, ...
  - Unisys U.S. LZW Patent No. 4,558,302 expired on June 20, 2003 [http://www.unisys.com/about\\_unisys/lzw](http://www.unisys.com/about_unisys/lzw)

# LZW

## Example

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

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		

**TABLE 8.7**  
LZW coding example.

- Exercise: verify that the dictionary can be automatically reconstructed during decoding. (G&W Problem 8.16)

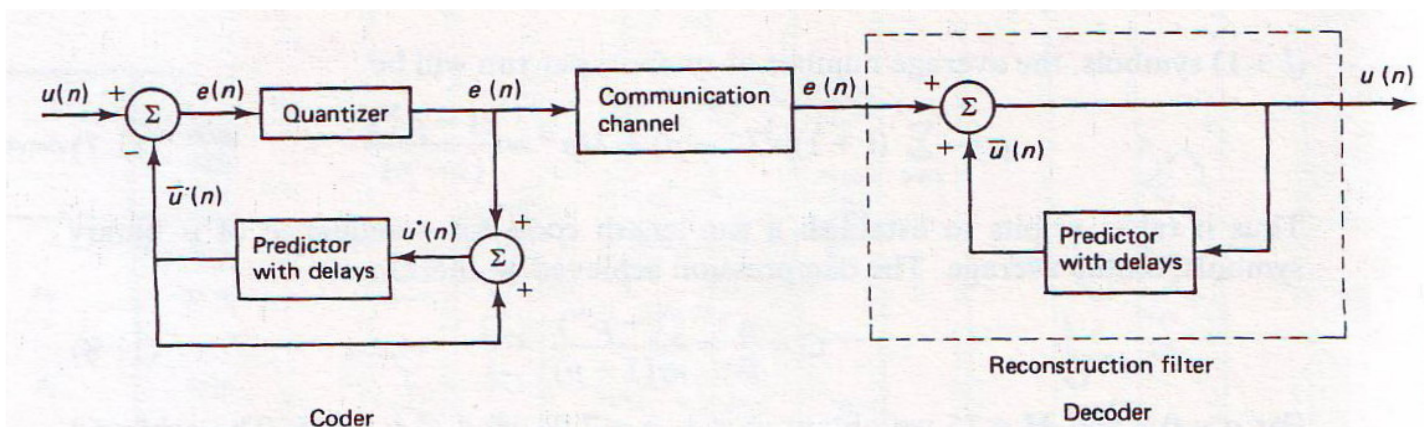
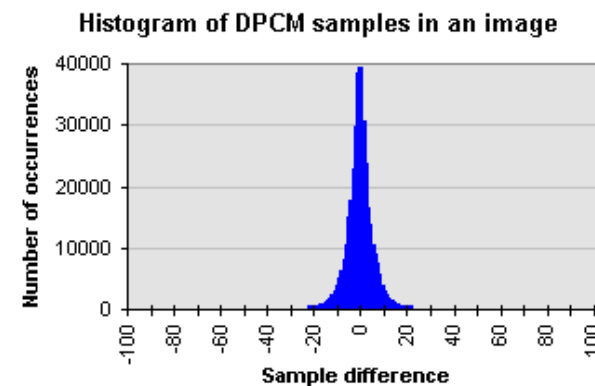
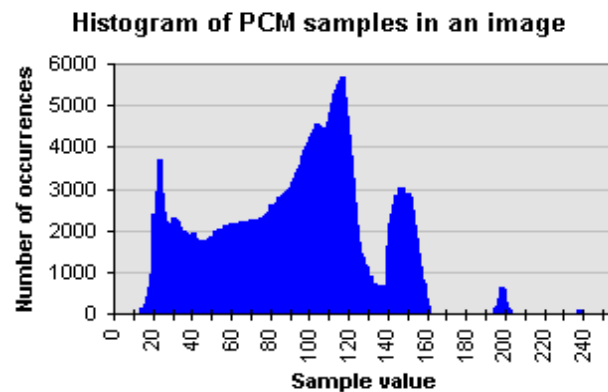
# coding symbol sequences

## Run-Length Coding

- Encode the number of consecutive '0's or '1's
- Used in FAX transmission standard
- Why is run-length coding with  $P(X=0) \gg P(X=1)$  actually beneficial?
  - See Jain Sec 11.3 (at course works)

# Predictive Coding

- Signals are correlated  $\rightarrow$  predict and encoding the difference lowers the bitrate
- Good prediction is the key: e.g. LPC (linear-predictive) speech coding



**Figure 11.5** Differential pulse code modulation (DPCM) CODEC

# Lecture Outline

# measuring image quality

## ■ Quality measures

### ■ PSNR (Peak-Signal-to-Noise-Ratio)

$$PSNR = 10 \log_{10} \left[ \frac{255^2}{\frac{1}{MN} \sum_{xy} |f'(x, y) - f(x, y)|^2} \right]$$

- Why would we prefer PSNR over SNR?

### ■ Visual quality

- Compression Artifacts
- Subjective rating scale

**TABLE 8.3**

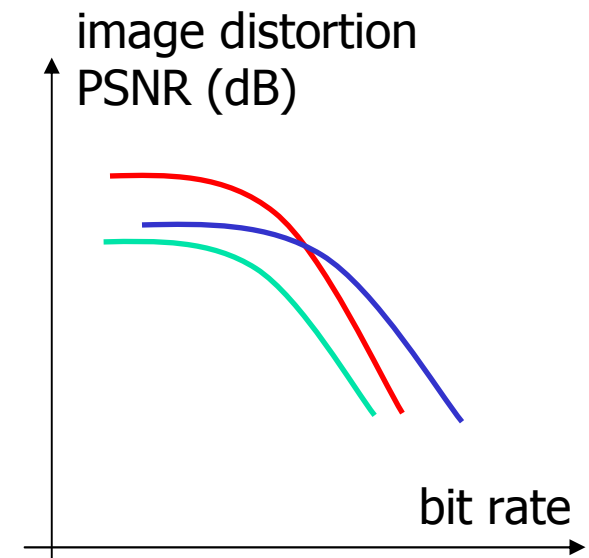
Rating scale of the  
Television  
Allocations Study  
Organization.  
(Frendendall and  
Behrend.)

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.



# measuring coding systems

- End-to-end measures of source coding system: Rate-Distortion
- Other considerations
  - Computational complexity
  - Power consumption
  - Memory requirement
  - Delay
  - Error resilience/sensitivity
  - Subjective quality



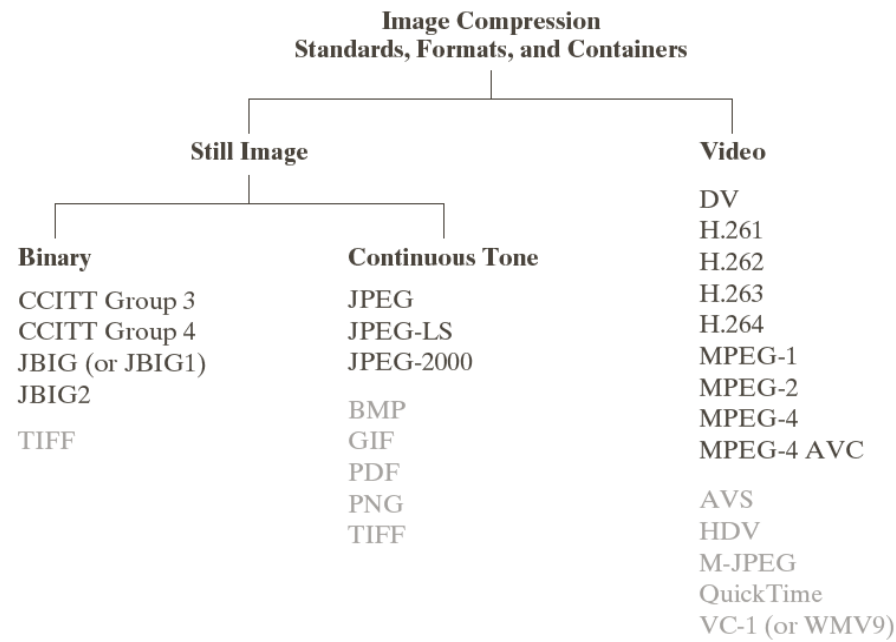
bpp: bit-per-pixel;

Kbps: Kilo-bits-per-second

# Image/Video Compression Standards

- Bitstream useful only if the recipient knows the code!
- Standardization efforts are important
  - Technology and algorithm benchmark
  - System definition and development
  - Patent pool management
- Defines the bitstream (decoder), not how you generate them (encoder)!

v • d • e Multimedia compression formats [hide]			
Video compression formats	ISO/IEC		ITU-T
	MPEG-1 · MPEG-2 · MPEG-4 ASP · MPEG-4/AVC		H.261 · H.262 · H.263 · H.264
Audio compression formats	ISO/IEC MPEG		ITU-T
	MPEG-1 Layer III (MP3) · MPEG-1 Layer II · AAC · HE-AAC		G.711 · G.722 · G.722.1 · G.722.2 · G.723 · G.723.1 · G.726 · G.728 · G.729 · G.729.1 · G.729a
Image compression formats	ISO/IEC/ITU-T		Others
	JPEG · JPEG 2000 · lossless JPEG · JBIG · JBIG2 · PNG · WBMP		APNG · ICER · MNG · BMP · GIF · ILBM · PCX · TGA · TIFF · HD Photo
Media container formats	General		Audio only
	3GP · ASF · AVI · DMF · DPX · FLV · Matroska · MP4 · MXF · NUT · Ogg · Ogg Media · QuickTime · RealMedia · VOB		AIFF · AU · WAV



**FIGURE 8.6** Some popular image compression standards, file formats, and containers. Internationally sanctioned entries are shown in black; all others are grayed.

## Digital TV Patent License Fees to Go to Columbia Very Soon

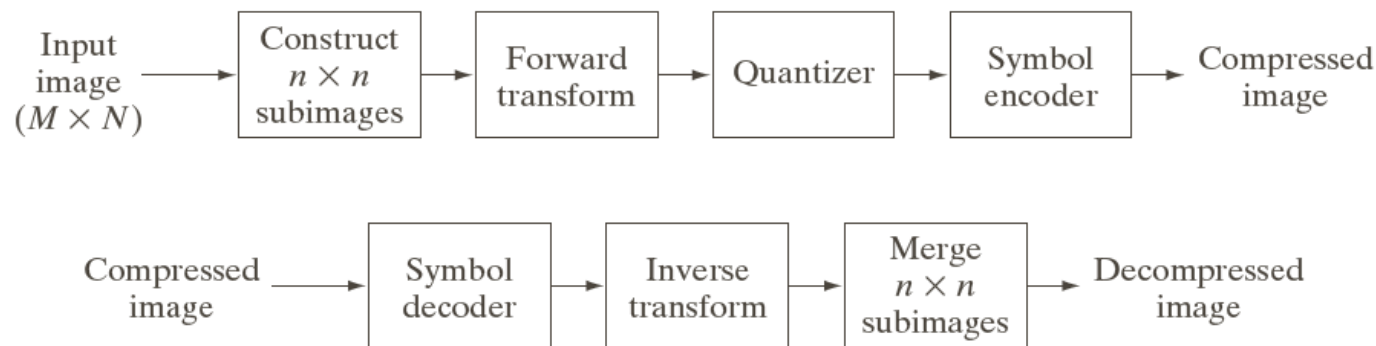
By Bob Nelson

**D**igital television is on its way and Columbia, the only academic institution in the patent pool created to license the MPEG-2 digital video compression standard, expects to begin receiving license fees from the technology as early as this year.

Columbia and eight companies together hold 33 patents that now comprise MPEG-2, which allows the transmission of high-quality video and audio signals over limited bandwidth. Dimitris Anastassiou, professor of electrical engineering at Columbia's School of Engineering and Applied Science and director of the Columbia New Media Technology Center, developed one of the MPEG-2 compression technologies with one of his graduate students.

"We believe the patent pool approach offers Columbia an excellent opportunity to receive significant royalty payments over the next few years," said Jack Granowitz, executive director of the Columbia Innovation Enterprise (CIE), the University's technology licensing office. Granowitz, along with

# block-based transform coding systems



a  
b

**FIGURE 8.21**

A block transform coding system:  
(a) encoder;  
(b) decoder.

- Review: properties of unitary transforms/DCT
  - De-correlation: highly correlated input elements  $\rightarrow$  quite uncorrelated output coefficients
  - Energy compaction: many common transforms tend to pack a large fraction of signal energy into just a few transform coefficients
- Symbol coding/decoding
  - predictive coding
  - run-length coding
  - Huffman codes
  - adaptive arithmetic coding ...

# JPEG compression standard (early 1990s)

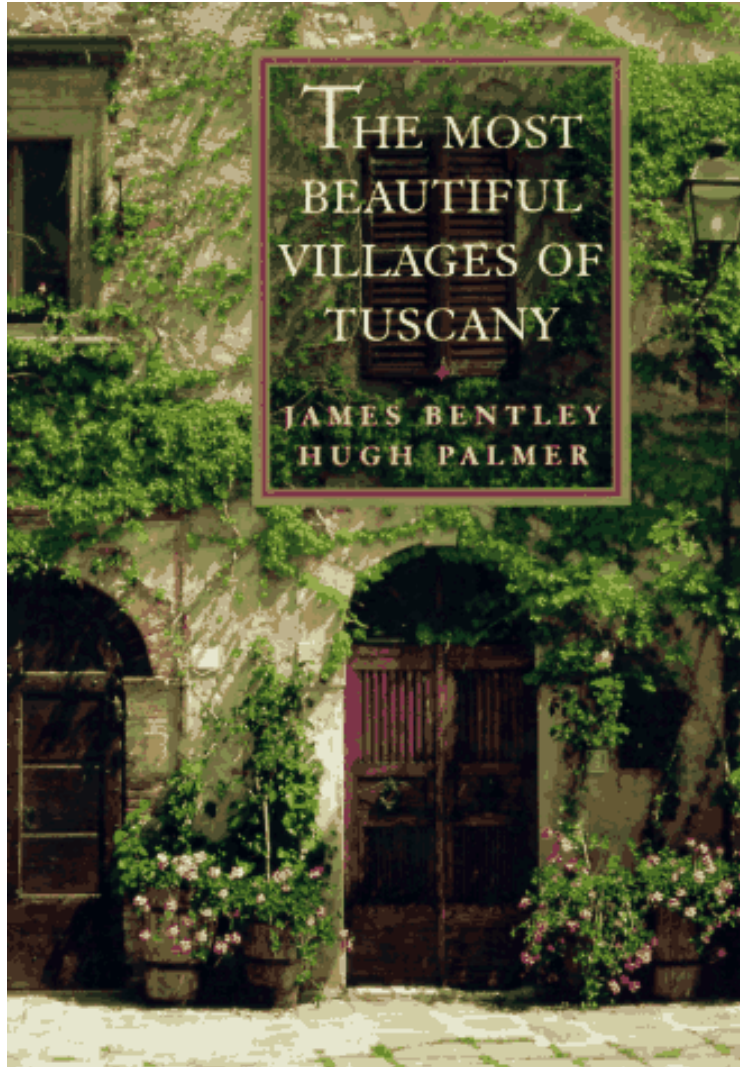
38

- JPEG - Joint Photographic Experts Group
  - Compression standard of generic continuous-tone still image
  - Became an international standard in 1992
- Allow for lossy and lossless encoding of still images
  - Part-1 DCT-based lossy compression
    - average compression ratio 15:1
  - Part-2 Predictive-based lossless compression
- Sequential, Progressive, Hierarchical modes
  - Sequential: encoded in a single left-to-right, top-to-bottom scan
  - Progressive: encoded in multiple scans to first produce a quick, rough decoded image when the transmission time is long
  - Hierarchical: encoded at multiple resolution to allow accessing low resolution without full decompression

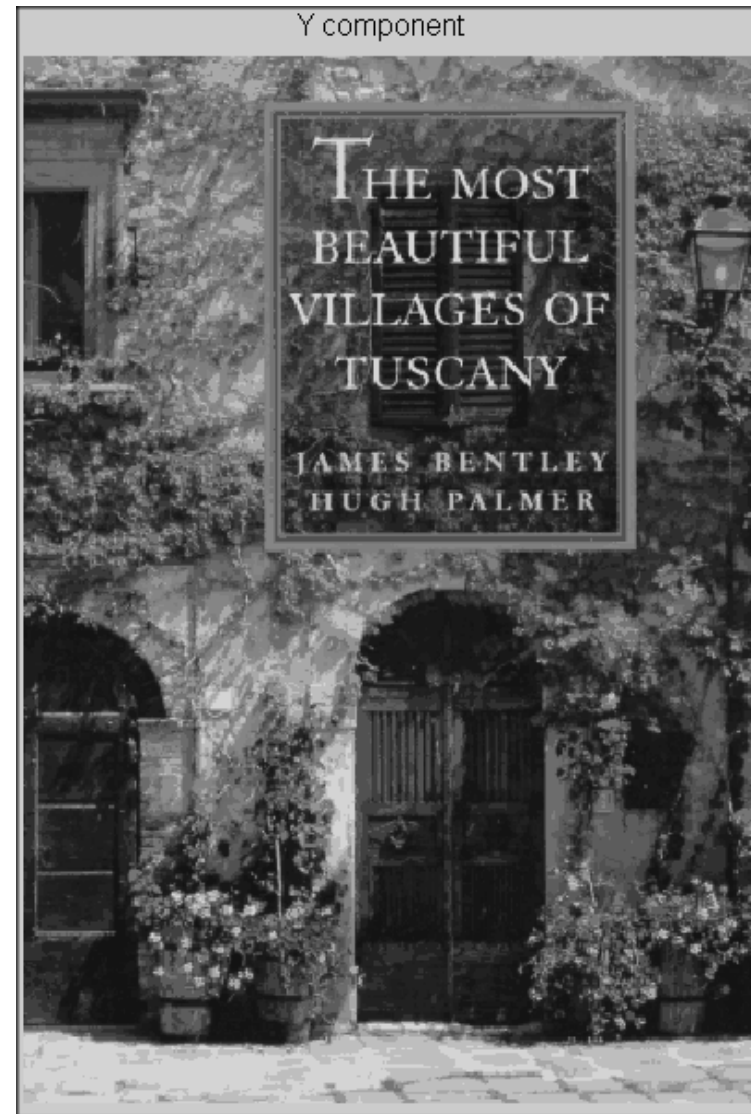
# Representation in JPEG

39

From Liu's EE330 (Princeton)



475 x 330 x 3 = 157 KB

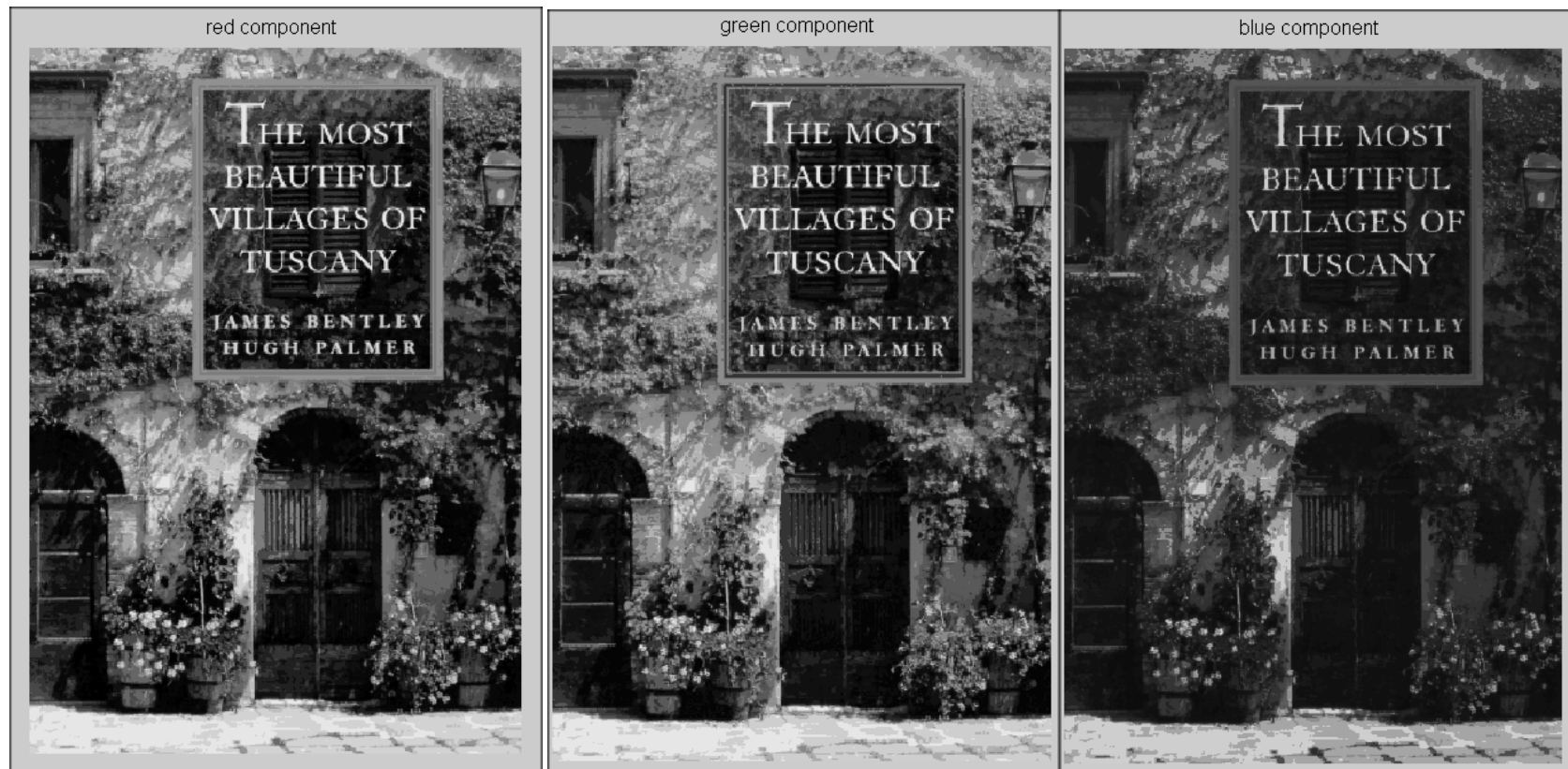


luminance



# RGB Components

40



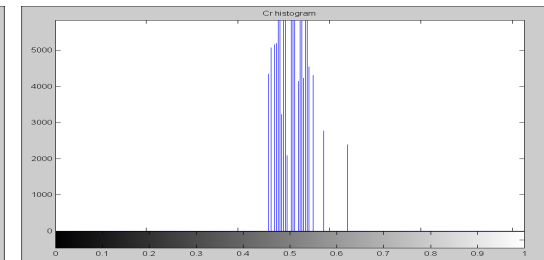
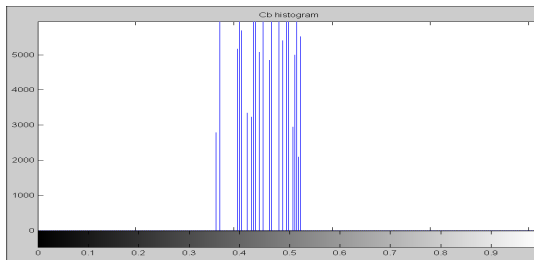
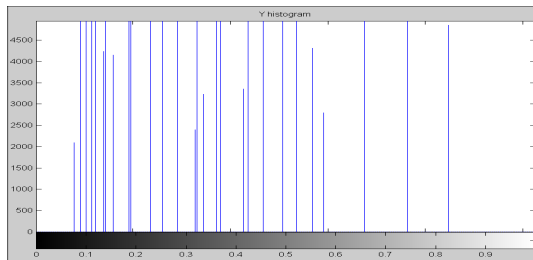
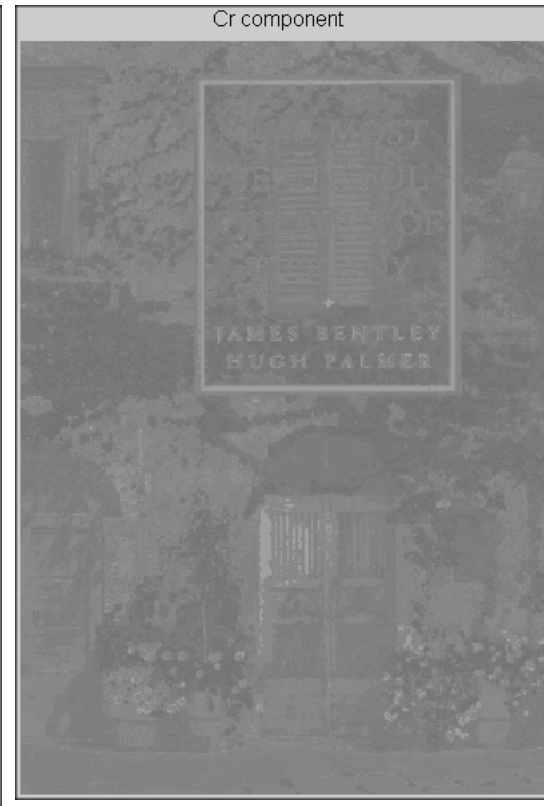
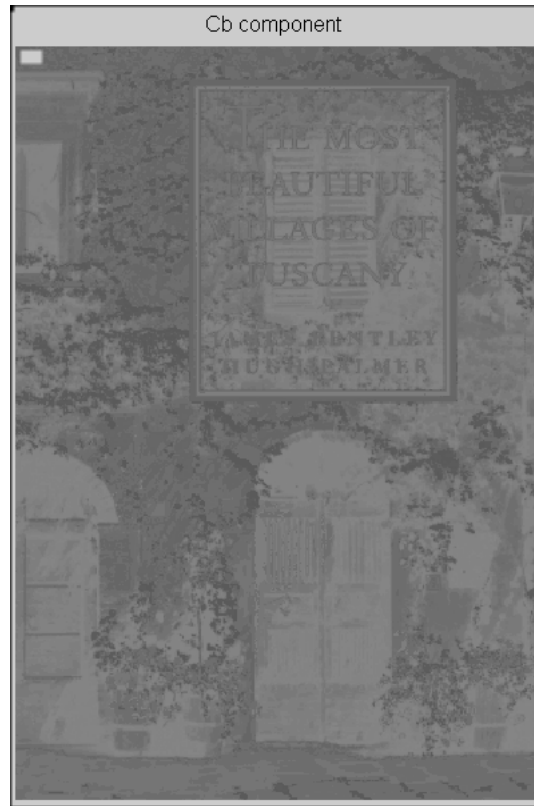
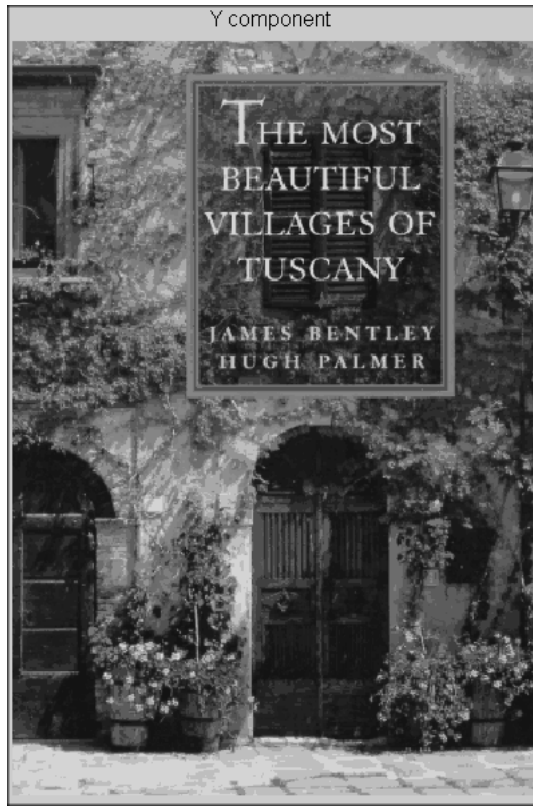
From Liu's EE330 (Princeton)



# Y U V (Y Cb Cr) Components

41

From Liu's EE330 (Princeton)



*Assign more bits to Y, less bits to Cb and Cr*

# baseline JPEG algorithm

42

- “Baseline”
  - Simple, lossy compression
    - Subset of other DCT-based modes of JPEG standard
- A few basics
  - 8x8 block-DCT based coding
  - Shift to zero-mean by subtracting 128 → [-128, 127]
    - Allows using signed integer to represent both DC and AC coeff.
  - Color (YCbCr / YUV) and downsample
    - Color components can have lower spatial resolution than luminance
  - Interleaving color components

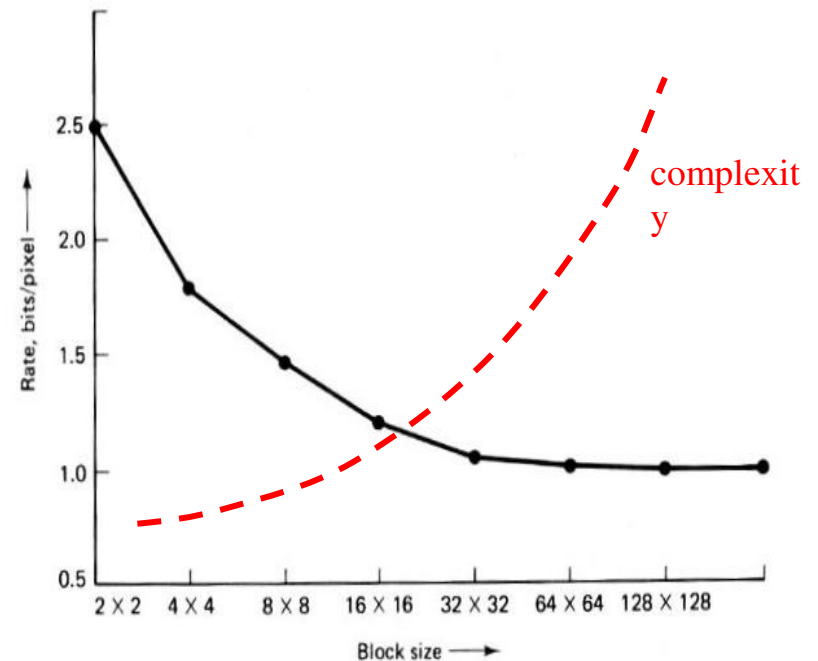
$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

(Based on Wang's video book Chapt.1)

# block-based transform

43

- Why block based?
  - High transform computation complexity for larger blocks
    - $O(m \log m \times m)$  per block in transform for  $(MN/m^2)$  blocks
  - High complexity in bit allocation
  - Block transform captures local info
- Commonly used block sizes: 8x8, 16x16, 8x4, 4x8 ...

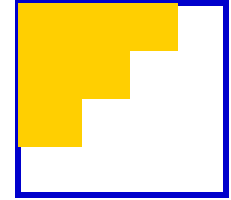


**Figure 11.16** Rate achievable by block KL transform coders for Gaussian random fields with separable covariance function,  $\rho = \rho_2 = 0.95$ , at distortion  $D = 0.25\%$ .

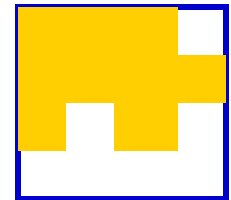
From Jain's Fig.11.16

# zonal coding and threshold coding

44



- Zonal coding
  - Only transmit a small predetermined zone of transformed coeff.



- Threshold coding
  - Transmit coeff. that are above certain thresholds

- Compare
  - Threshold coding is inherently adaptive
    - introduce smaller distortion for the same number of coded coeff.
  - Threshold coding needs overhead in specifying index of coded coeff.
    - run-length coding helps to reduce overhead

# perform quantization

- Input:
  - 8x8 DCT image  $X(u,v)$
  - Quantization table  $Q(u,v)$
- The quantizer output is:
 
$$I(u,v) = \text{Round}[X(u,v)/Q(u,v)]$$
  - “round” is to the nearest integer
- JPEG default luminance table shown on the right
  - Smaller  $Q(u,v)$  means a smaller step size and hence more resolution, vice-versa
  - $Q(u,v)$  may be scaled by a quality factor

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

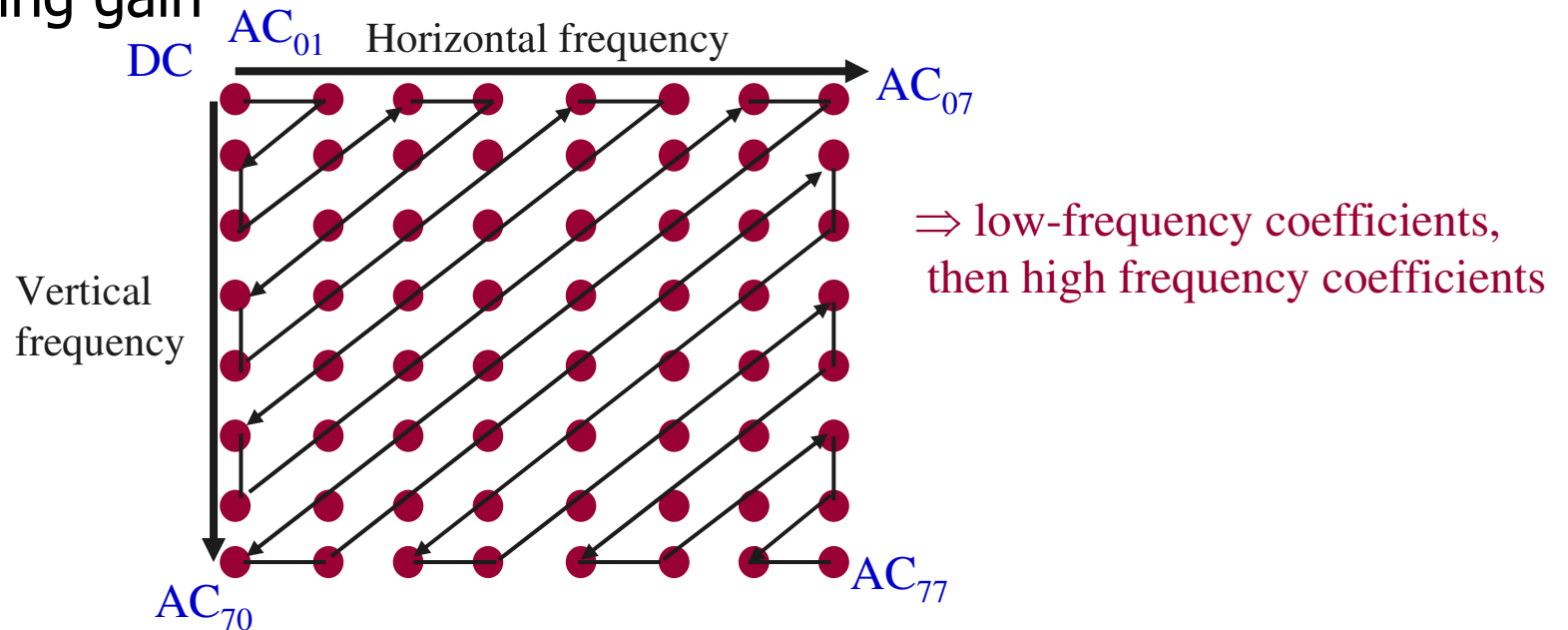
# quantization of transform coefficients

- Default quantization table
  - “Generic” over a variety of images
- Adaptive Quantization (bit allocation)
  - Different quantization step size for different coeff. bands
  - Use same quantization matrix for all blocks in one image
  - Choose quantization matrix to best suit the image
  - Different quantization matrices for luminance and color components
- Quality factor “Q”
  - Scale the quantization table
  - Medium quality  $Q = 50\% \sim$  no scaling
  - High quality  $Q = 100\% \sim$  unit quantization step size
  - Poor quality  $\sim$  small  $Q$ , larger quantization step
    - visible artifacts like ringing and blockiness



# encode an image block

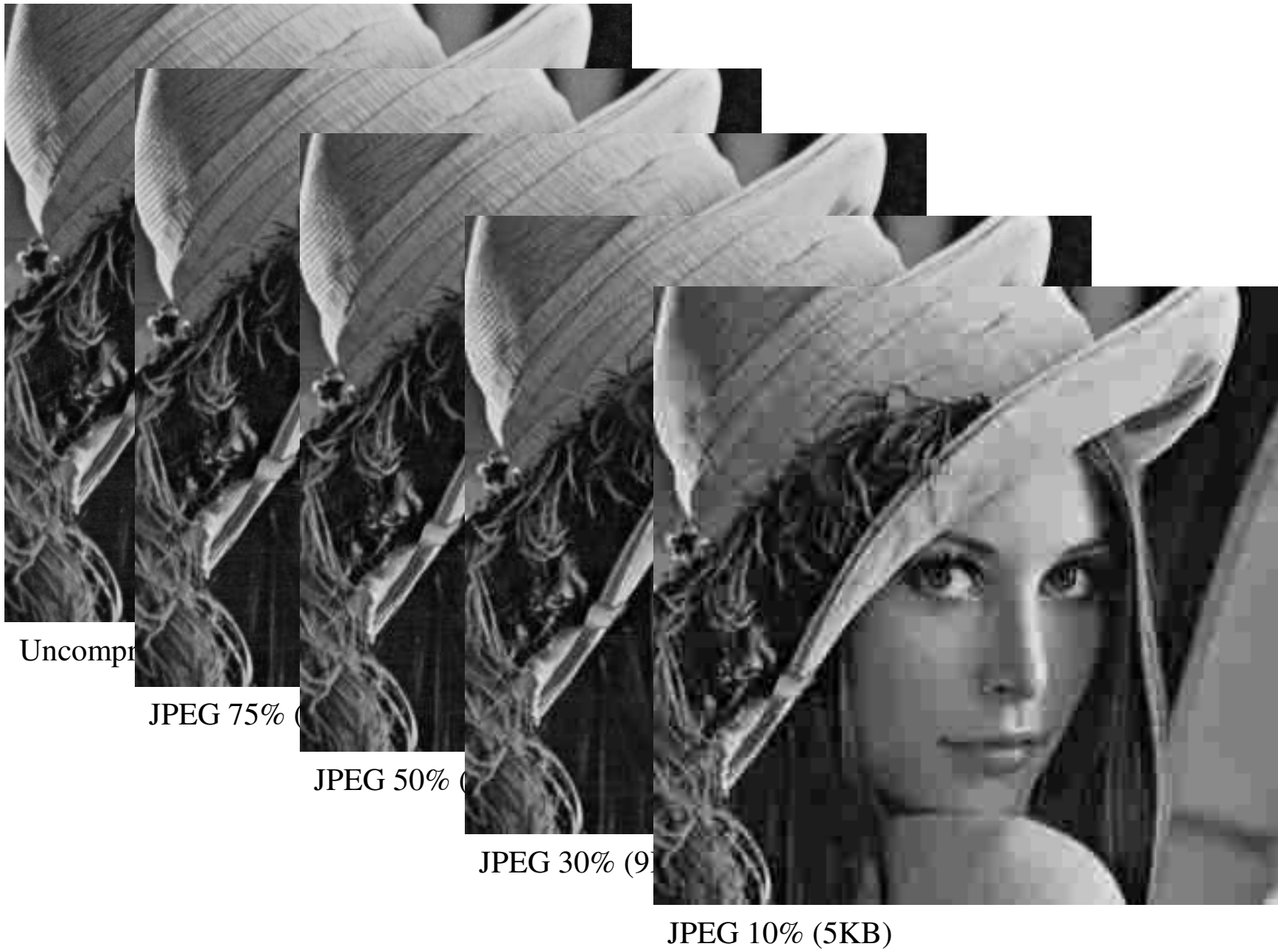
- Basic tools
  - Run-length coding
  - Predictive coding (esp. for DC coefficient)
  - Entropy coding (Huffman, etc.)
- Scan order
  - zig-zag scan for block-DCT to better achieve run-length coding gain



# encoding a block in JPEG

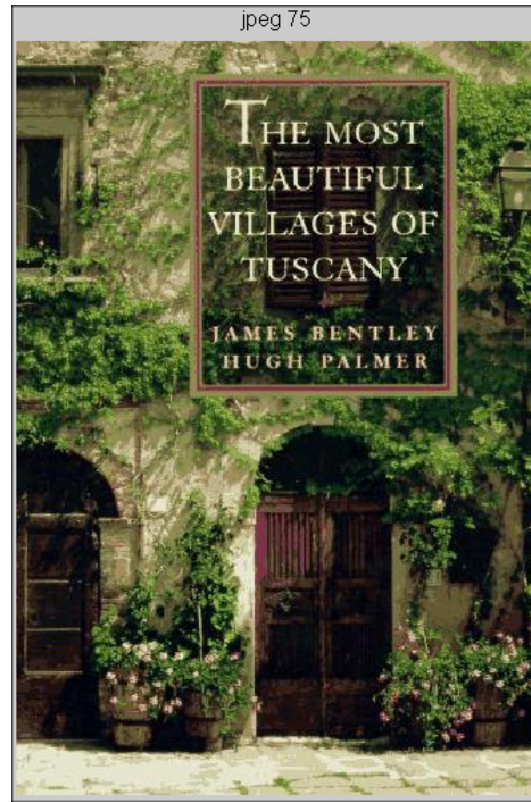
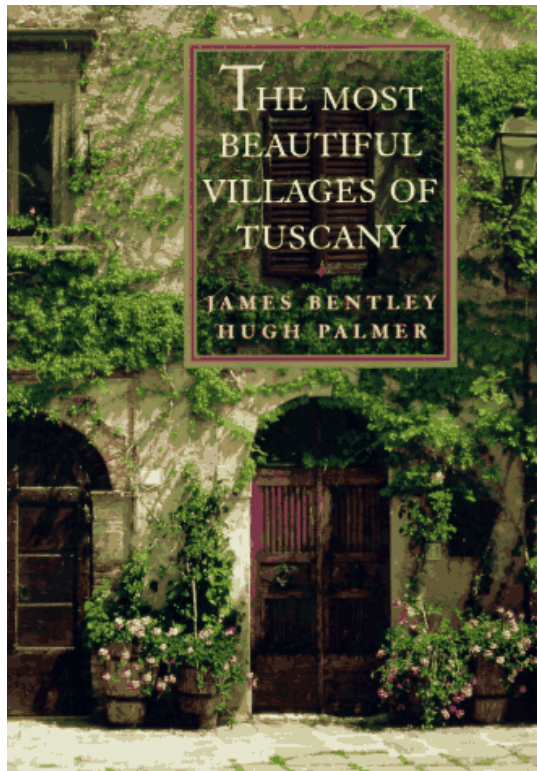
- Differentially encode DC (and quantize)
    - ( SIZE, AMPLITUDE ), with amplitude range in [-2048, 2047]
  - AC coefficients in one block
    - Zig-zag scan after quantization for better run-length
      - save bits in coding consecutive zeros
    - Represent each AC run-length using entropy coding
      - use shorter codes for more likely AC run-length symbols
    - Symbol-1: ( RUNLENGTH, SIZE ) → Huffman coded
    - Symbol-2: AMPLITUDE → Variable length coded
- RUNLENGTH  $\in [0,15]$   
                   # of consecutive zero-valued AC coefficients  
                   preceding the nonzero AC coefficient  $\in [0,15]$
- SIZE  $\in [0 \text{ to } 10 \text{ in unit of bits}]$   
                   # of bits used to encode AMPLITUDE
- AMPLITUDE  $\in$  in range of [-1023, 1024]



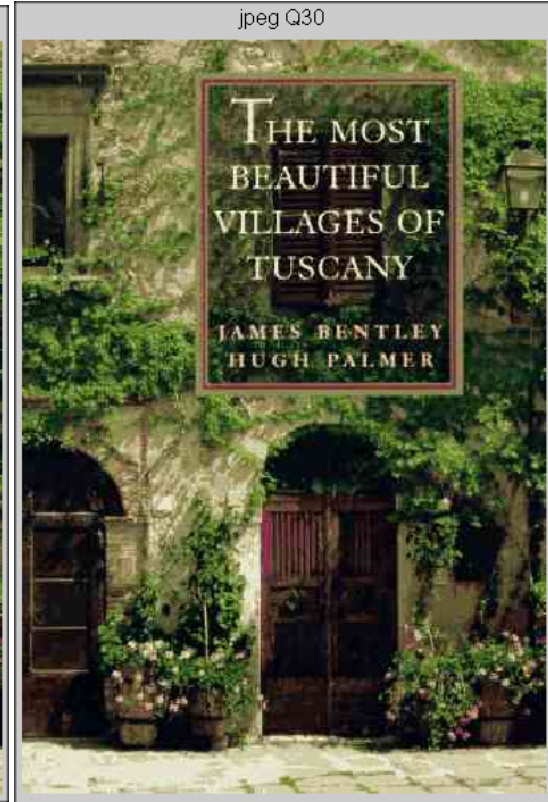


# JPEG Compression (Q=75 & 30)

50

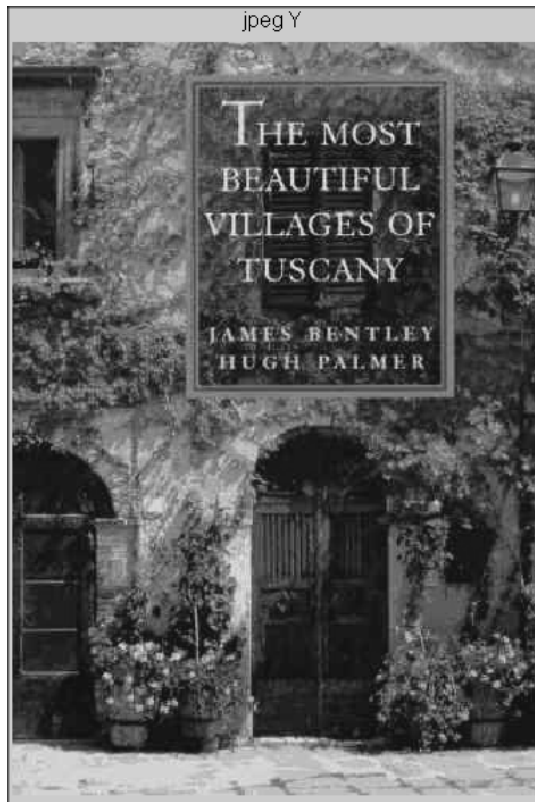


45 KB



22 KB

# Y Cb Cr After JPEG (Q=30%)



# JPEG 2000

- Better image quality/coding efficiency, esp. low bit-rate compression performance
  - DWT
  - Bit-plane coding (EBCOT)
  - Flexible block sizes
  - ...
- More functionality
  - Support larger images
  - Progressive transmission by quality, resolution, component, or spatial locality
  - Lossy and Lossless compression
  - Random access to the bitstream
  - Region of Interest coding
  - Robustness to bit errors

# Video ?= Motion Pictures

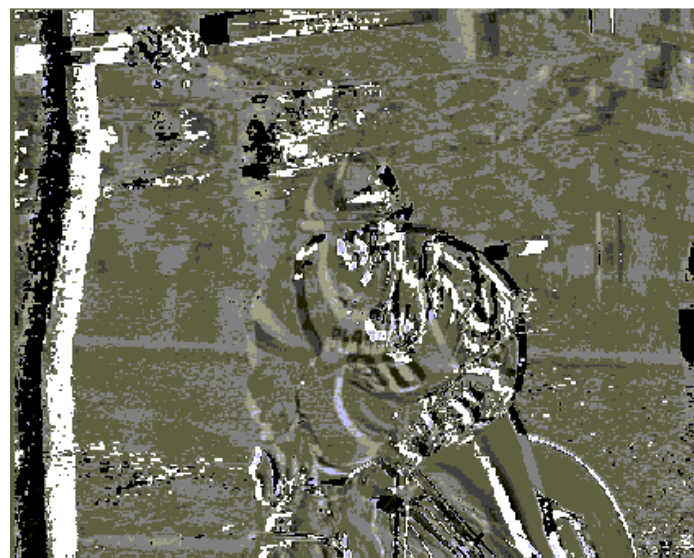
53

- Capturing video
    - Frame by frame => image sequence
    - Image sequence: A 3-D signal
      - 2 spatial dimensions & time dimension
      - continuous  $I(x, y, t)$  => discrete  $I(m, n, t_k)$
  - Encode digital video
    - Simplest way ~ compress each frame image individually
      - e.g., “motion-JPEG”
      - only spatial redundancy is explored and reduced
    - How about temporal redundancy? Is differential coding good?
      - Pixel-by-pixel difference could still be large due to motion
- ➔ Need better prediction





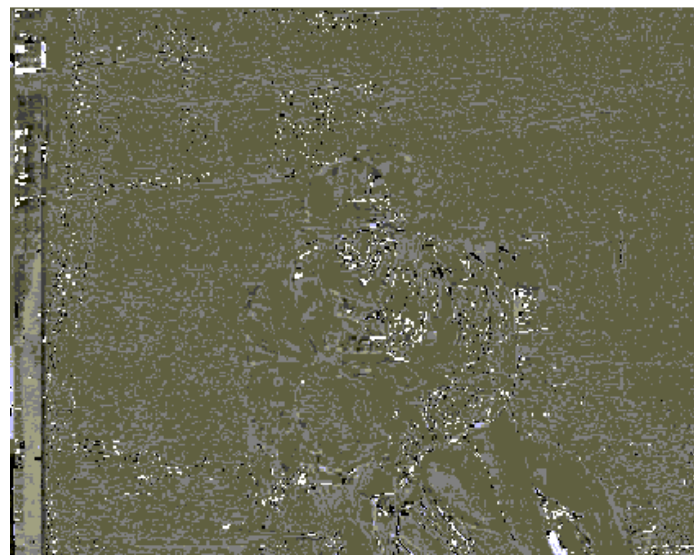
"Horse ride"



Pixel-wise difference w/o motion compensation

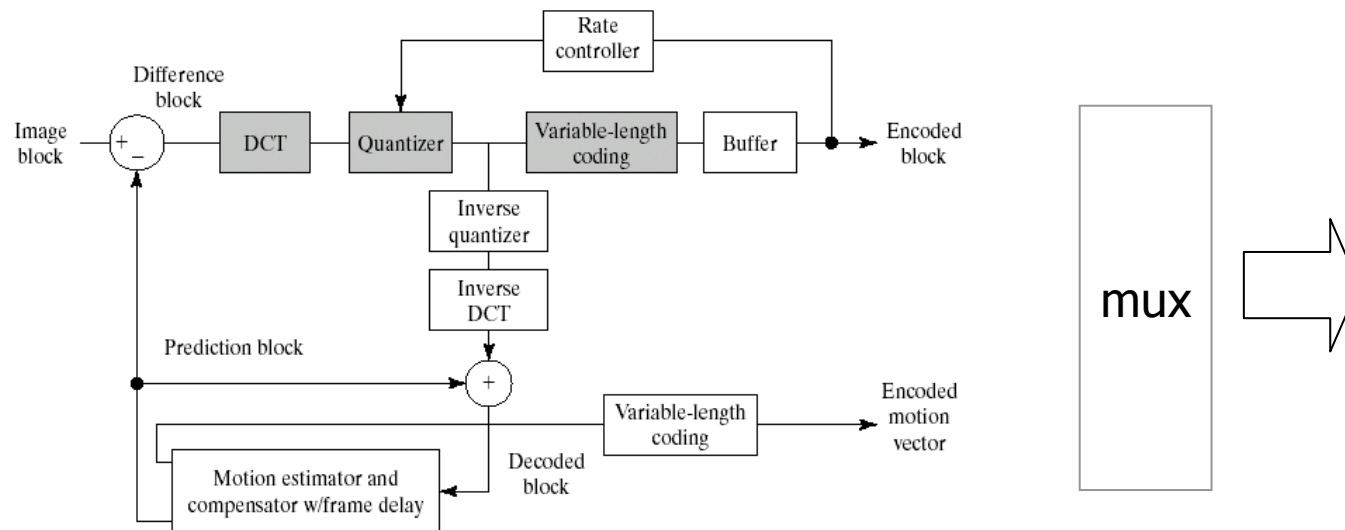


Motion estimation

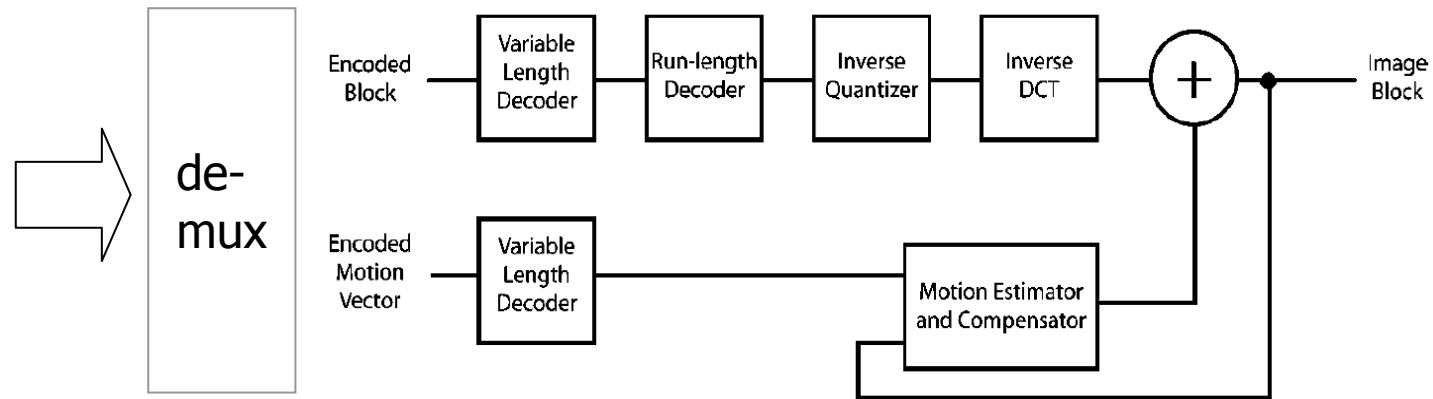


Residue after motion compensation

# hybrid video coding system

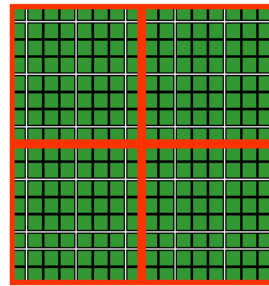


**FIGURE 8.47** A basic DPCM/DCT encoder for motion compensated video compression.

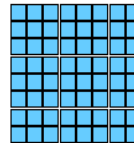


## a ideas in video coding systems

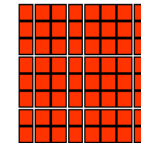
- Work on each macroblock (MB) (16x16 pixels) independently for reduced complexity
  - Motion compensation done at the MB level
  - DCT coding at the block level (8x8 pixels)



4 8x8 Y blocks



1 8x8 Cb blocks

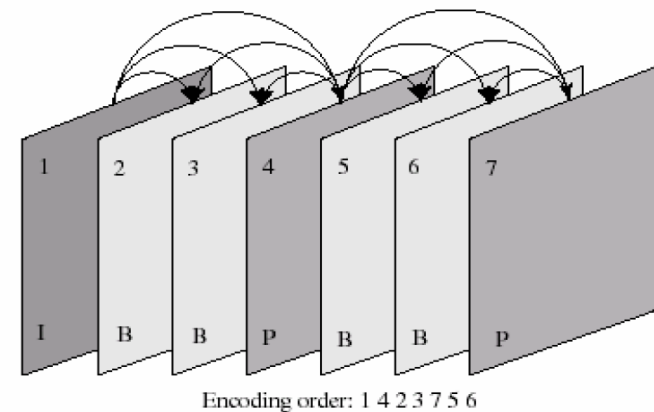
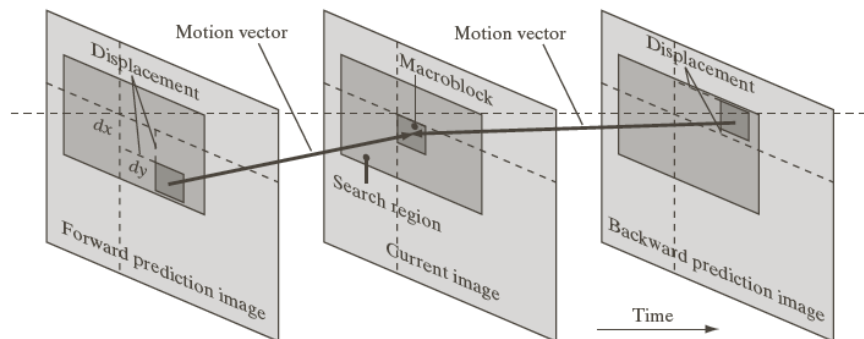


1 8x8 Cr blocks



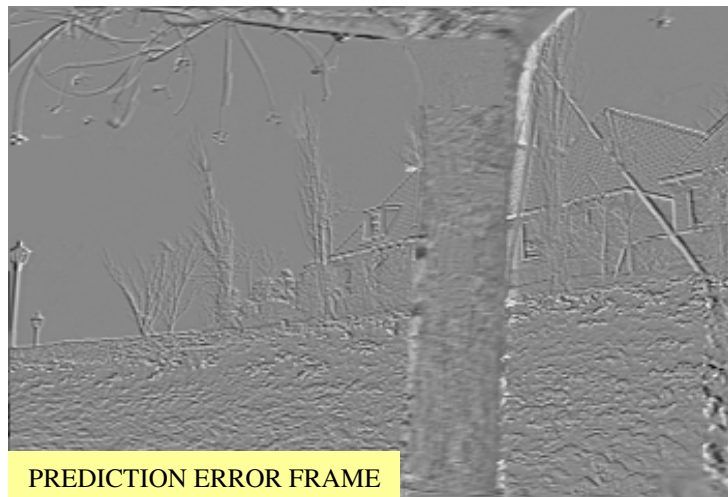
# representing motion

- Predict a new frame from a previous frame and only code the prediction error --- *Inter* prediction on "B" and "P" frames
- Predict a current block from previously coded blocks in the same frame --- *Intra* prediction (introduced in the latest standard H.264)
- Prediction errors have smaller energy than the original pixel values and can be coded with fewer bits
  - DCT on the prediction errors
- Those regions that cannot be predicted well will be coded directly using DCT --- Intra coding without intra-prediction



# motion compensation

- Help reduce temporal redundancy of video



# motion estimation

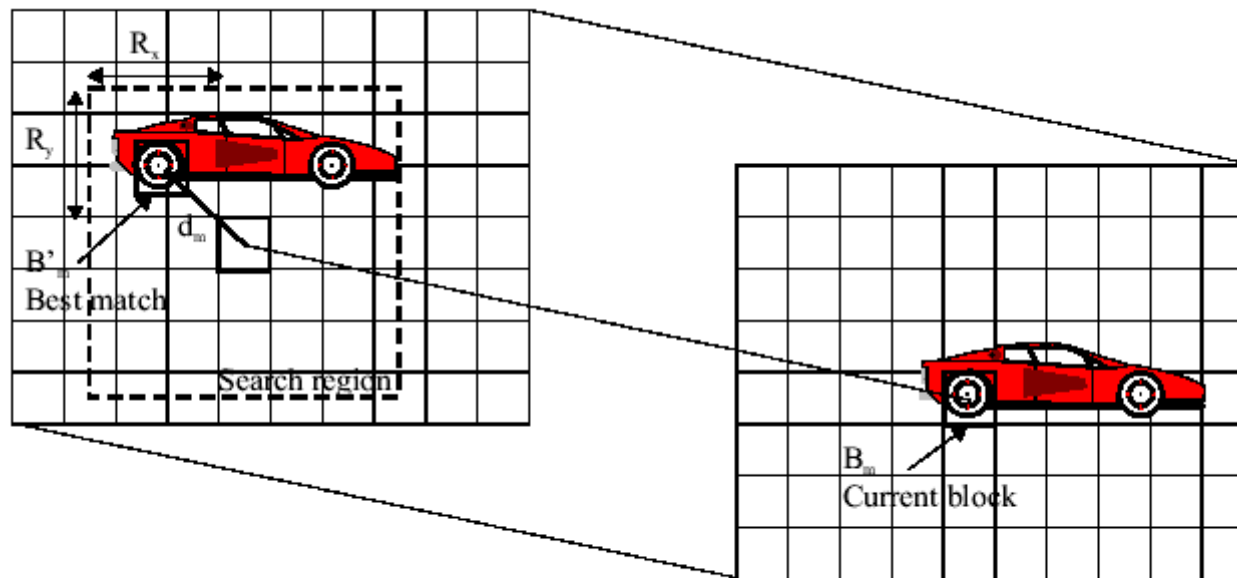
59

- Help understanding the content of image sequence
- Help reduce temporal redundancy of video
  - For compression
- Stabilizing video by detecting and removing small, noisy global motions
  - For building stabilizer in camcorder
- A hard problem in general!

# block-matching with exhaustive search

60

- Assume block-based translation motion model
- Search every possibility over a specified range for the best matching block
  - MAD (mean absolute difference) often used for simplicity



**Figure 6.6.** The search procedure of the exhaustive block matching algorithm.

From Wang's  
Preprint Fig.6.6

## audio coding versus image coding

	MP3 (wideband audio coding)	JPEG
Data Unit	Frame	Block
Transform	MDCT	DCT
Quantization	Fixed Quantization matrix base on psychoacoustic masking	Baseline quantization matrix + adaptive rate control
Entropy coding	Huffman code	Huffman code, run-length, differential

# VC demo



**[I,C]<sup>T</sup>**

Information and Communication Theory Group



**TU Delft**

Delft University of Technology

[Home](#) | [Contact](#) |

---

**ICT - Navigator**

- ☐ Home
- ☐ People
- ☐ Research Themes
- ☐ Publications
- ☐ **Software**
  - ☐ VcDemo
    - ☐ Download and Problems
    - ☐ Screen Shots
    - ☐ Modules
    - ☐ History
  - ☐ PRTTools
  - ☐ Genlab
  - ☐ Erlang/Engset Calculator
  - ☐ AFwRQIM
  - ☐ GenGam
  - ☐ VfmDirect
- ☐ Education
- ☐ Vacancies
- ☐ Students
- ☐ Internal
- ☐ ICTWiki
- ☐ Webmail
- ☐ Events
- ☐ Contact

---

**Login Form**

Username

Password

[Password forgotten?](#)

[Home](#) > [Software](#) > **VcDemo**

---

## Image and Video Compression Learning Tool VcDemo

**VcDemo is an interactive image and video compression free-ware software package for Windows.**

It is intended as a tool for learning about compression techniques: from basic sampling and PCM to today's most advanced scalable embedded zerotree wavelet image compression technique, and the MPEG video compression standard.

*The latest VcDemo release is Version 5.03 (September 2004).*

VcDemo is an ideal tool for students to explore the possibilities of compression theory on real images using algorithms that text books teach about. The package is also very suitable for on-line demonstrations in lectures. A user manual (F1-help) is available, as well as a set of exercises that can serve as home work.

Download  
Now



The screen shot above gives an impression of the user interface. VcDemo is entirely menu-driven. No programming is necessary, but the user has control over the crucial compression parameters. Example of such parameters are bit rate, prediction structure in DPCM, the block-size in DCT compression, and the Group-of-Picture structure in MPEG.



# Recent Activities in Image Compression

## ■ Build better, more versatile systems

- High-definition IPTV
- Wireless and embedded applications
- P2P video delivery

## ■ In search for better basis

- Curvelets, contourlets, ...

## ■ “compressed sensing”



### Compressed Sensing Research at Rice University

Overview, References, and Software	Compressed Sensing Research at Rice	
Compressive Imaging	Connections to Dimensionality Reduction	Tree-Matching Pursuit
Analog-to-Information Conversion	Connections to Information Theory	The Rice Team
Compressive Signal Processing	Multi-Signal and Distributed Compressed Sensing	

#### Compressive Imaging

We are also developing algorithms and hardware to support a new theory of *Compressive Imaging*. Our approach is based on a **new camera** that directly acquires random projections of a digital image/video without first collecting the set of pixels. Our camera architecture employs a digital micromirror array to perform optical calculations of linear projections of an image onto pseudorandom binary patterns. Its hallmarks include the ability to obtain an image with a single detection element (photodiode) while sampling the image fewer times than the number of pixels. This camera also inherits the compressed sensing benefits of universality, robustness, progressivity, and computational asymmetry. Perhaps the most intriguing feature of the system is that, since it relies on a single photon detector, it can be adapted to image at wavelengths that are currently impossible with conventional CCD and CMOS imagers.

Publications: *SPIE Electronic Imaging* (2006), *ICIP Conference* (2006).

The diagram illustrates the hardware architecture for Compressive Imaging. It shows an "Image encoded by DMD and random basis" being projected onto a "Low-cost, fast, sensitive optical detection" unit. This unit includes a "PD" (Photodiode) and an "A/D" (Analog-to-Digital) converter. The output is "Compressed, encoded image data sent via RF for reconstruction". The data is received by an "Rcvr" (Receiver) and then processed by a "DSP" (Digital Signal Processor). A "DMD" (Digital Micromirror Device) and an "RNG" (Random Number Generator) are also shown as part of the system.

#### Analog-to-Information Conversion

We are currently developing new theory and mixed-signal hardware designs for *Analog-to-Information (A/I) Conversion*. Such systems will significantly reduce the burden on analog-to-digital conversion for sampling and compression of high-bandwidth signals. The key idea is that, rather than sampling an analog signal at its Nyquist rate and then computing incoherent projections (effectively discarding most of the samples), these systems will directly acquire and produce a low-rate stream of incoherent measurements.

Rice / Michigan A/I Project  
Publication: *ICASSP Conference* (2006).

The block diagram shows the process of Analog-to-Information Conversion. It starts with an "analog signal  $x(t)$ " entering a block labeled "A/I  $\Phi$ ". The output of this block is "digital measurements  $y_n$ ". These measurements then enter a block labeled "DSP", which produces the final "information statistics".

# Summary

- The image/video compression problem
- Source coding
  - For i.i.d. symbols
  - For symbol streams
- Image/video compression systems
  - MPEG/JPEG and beyond
- Next time: reconstruction in medical imaging and multimedia indexing and retrieval