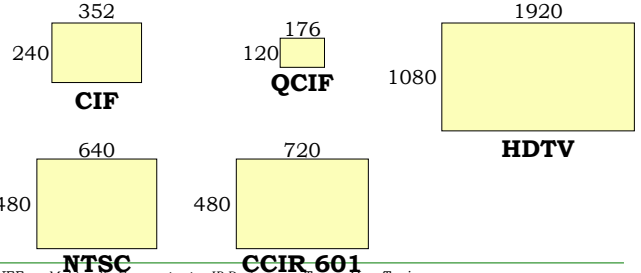
Multimedia System Realization

Outline

- Compression Fundamentals
- JPEG Coding Standards
- H.261 & H.263
- MPEG1 & MPEG2
- MPEG4
- JPEG 2000

Digital Video

- What's digital video?
  - Computer user ® Consumer user



Real-Time Requirement

Format	Data Amount	Bit-Rate
352x240 CIF	30.4 Mbits/s	1.5 Mbits/s
720x480 CCIR 601	165.9 Mbits/s	15-25 Mbits/s
1920x1080 HDTV	995.3 Mbits/s	80 Mbits/s

Why Compression? (Digital Video)

- Large storage requirements
  - encyclopedia
- Relatively slow storage devices
  - CD-ROM (300KB/sec transfer rate)
- Network's bandwidth
  - Ethernet, token ring (tens of Mb/sec)
  - ATM, FDDI (hundreds of Mb/sec)

Data Compression

- Example 1: Facsimile image transmission A4 page = 8.5 x 11 inches in 200dpi digitized to 3.74 Mbits for 14.4 kbits/s modem needs 5.62 minutes.
- Example 2: Video-based CD-ROM 30 fps 720 x 480 resolution generates date at 20.736 Mbytes/sec only 31 seconds of video be stored on 650MByte CD-ROM.

Storage Requirements for Multimedia applications

	TEXT	IMAGE	AUDIO	ANIMATION	VIDEO
OBJECT TYPE	-ASCII -EBCDIC	-Bit-mapped graphics -Still photos -Faxes	Non-coded stream of digitized audio or voice	Synched image and audio stream at 15-19 frames/s	TV analog or digital image with synched streams at 24-30 frames/s
SIZE AND BANDWIDTH	2 KB per page	-Simple (grayscale) 77KB per image (320x240x8bits) -Detailed (color) 3 MB per image (1100x900x24bits)	Voice/Phone 8KHz/8 bits- (mono) 8-44 KB/s Audio CD 44.1 KHz, 16 bits/stereo 176 KB/s (44.1KHz x 2 ch x 16 bits)	16 bit color, 15 frames/sec 6.5 MB/s (32 x640 x16bits x 15 frames/sec)	24 bit color, 30 frames/sec 27.6 MB/s (640X480X24bits x 30 frames/sec)

Table 4.1 Storage requirements for various media types.

Image Compression

- **Purposes:** Reduce the number of bits needed to represent an image with reasonably preserved quality for a given applications.
- It is used to
  - Reduce memory size needed for image storage
  - Reduce bandwidth/time requirement for image transmission
- How? Answer: Remove Redundancy and Irrelevancy
  - **Redundancy:** regarding to the statistical property of an image, usually exploited by entropy coding.
  - **Irrelevancy:** regarding to the redundancy invisible or hard to be seen by an observer, which depends on viewing conditions and is usually exploited by transform/quantization and human visual system (HVS).

Considerations for Compression

- Picture quality vs. bitrate
- Variable bit rate vs. constant bit rate
- Robustness: noisy channels
- Interactivity: algorithm that operates on a small groupe of pels
- Compression and packetization delay: more efficient algorithm introduces more compression and packetization delay

Considerations for Compression (cont'd)

- Multiple encoding: higher quality is required for multiple codings
- Symmetry: the analysis phase of encoding makes the encoder more expensive
- Scalability: different resolutions (in space, time, amplitude, ...)
  - algorithms with highest compression efficiency usually are not very scalable

Data Compression Coding Basis

- Achieve high compression performance while keep good picture quality
- Theorem:
  - Spatial redundancy - DPCM, DCT, DFT, Subband, Wavelet
  - Temporal redundancy - DPCM, MC/ME
  - Statistical redundancy - RLC, VLC
  - Perceptual redundancy - VQ, fractal

Compression Technology

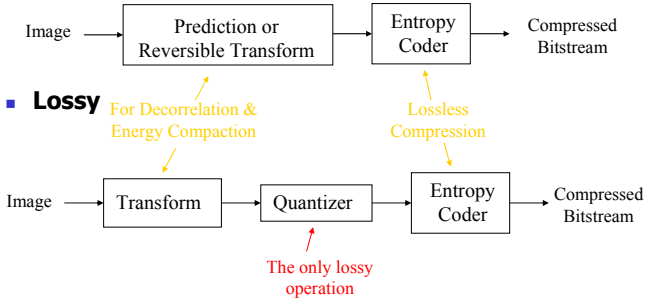
- Remove redundancy
  - Statistic (Entropy coding)
  - Spatial (Transform coding)
  - Temporal (Motion Estimation)
- Characteristics of video
  - Statistical redundancy
    - lossless
    - depend on the probabilistic characterization of signal
  - Perceptual redundancy
    - lossy, irreversible
    - complex, depends on context and application

Types of Image Compressing

- **Lossless** (Reversible, Bit Preserving):
  - The decoded image is identical to the original
  - Only the statistical redundancy is exploited
  - Compression ratio is typically about 2:1 to 3:1
- **Lossy** (Irreversible):
  - The reconstructed image is not exactly the same as the original
  - Both the statistical redundancy and the perceptual irrelevancy of the image are exploited.
  - Much higher compression ratio can be achieved.
  - **Visually Lossless** is often used to refer to lossy compression that results in no visible degradation.

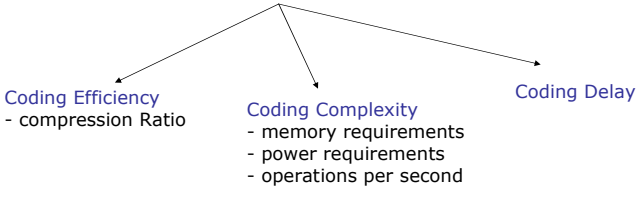
Image Compression Framework

- **Lossless:**

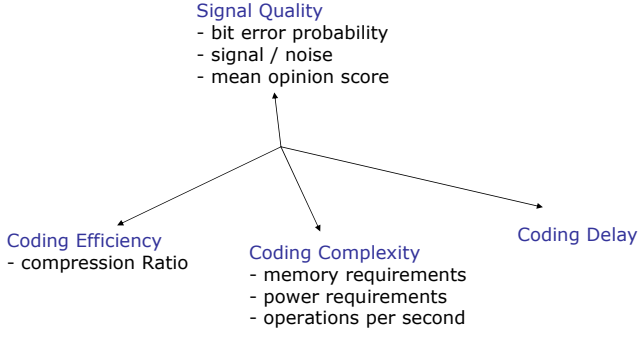


- **Lossy**

Tradeoffs in Lossless Compression



Tradeoffs in Lossy Compression



Lossless Compression

- **Definition:** Compression methods for which the original uncompressed data set can be recovered exactly from the compressed stream
- **Usage**: digital medical data , bitonal image, artwork design, ...

Preliminaries

- **Generic model:** given an input set of symbols, a modeler generates an estimate of the probability distribution of the input symbols. This probability model is then used to map symbols into codewords.
- **Entropy coding:** the combination of the modeling and the symbol-to-codeword mapping functions is usually referred to as "Entropy coding"
- **Key idea:**
  - short codewords for symbols occurred with high probability long codewords for symbols occurred with low probability
- **Entropy decoding:** decompression process

Measurements of Performance

- Complexity
  - how fast the algorithm performs
- Memory required
- Amount of compression
  - compression ratio = length of the original data / length of the compressed data
  - rate = bits per sample (bpp: bits per pixel)
    or bits per second (64kb/s)
- How closely the reconstruction resembles the original (for lossy compression)
  - objective measure
- SNR (signal-to-noise ratio)
  - subjective measure
- MOS (Mean Opinion Score)

Lossy compression

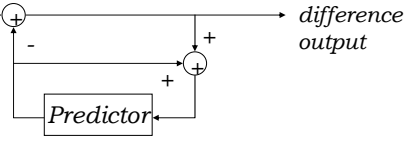
- The decompression yields an imperfect reconstruction of the original image data.
- Given the level of image loss (or distortion) D, there is always a bound on the minimum bit rate of the compressed bit stream.
- A common measure for *D* is the ***mean square error*** between the encoded and decoded images, normalized by the variance of the input signal

Entropy Coding

- Usually includes **statistical models** of the input data.
  - The models can be **fixed** or **adaptive**.
  - **Context Model**: use the neighboring context information to switch models.
- **Lossless compress** the input data by, e.g.,
  - Run-length Coding, 2D or 3D symbol coding
  - Lempel-Ziv-Welch (LZW) coding
  - Huffman Coding, Golomb-Rice coding
  - Quad-tree coding
  - Arithmetic coding

Differential Pulse Code Modulation

- Highly correlated on adjacent pixels
- Using the value x_{i-1} to predict the next value x_i



Ex.

- original pixel: 192 188 189 193 200 ....
- difference data: .... -4 1 4 7 ....

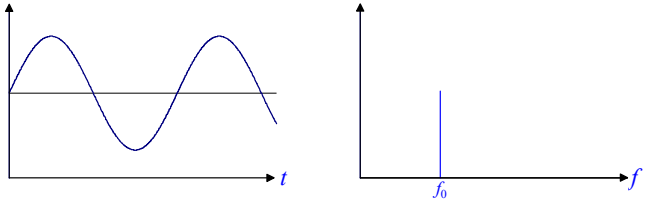
Transform or Prediction

- A **reversible** process (or often near-reversible, due to finite precision arithmetic) that provides a decorrelated and energy-compacted representation of an image.
- **Decorrelation:** Decorrelate data to make them amenable to efficient entropy coder with low-order models.
- **Energy Compaction:** Redistribute energy to relatively few coefficients and thus make it easy to remove redundancy.
- **Examples:**
  - DPCM, Predictive Coding
  - Discrete Cosine Transform (DCT), Karhunen-Loeve Transform (KLT)
  - Wavelet Transform, Subband Decomposition
  - Color space transformation (e.g., RGB to YIQ)

Spatial Redundancy

- Intra-frame Coding(I-frame)
- Block-based schemes(Transform Coding)
  - De-correlation of transform coefficients
  - Energy concentration
  - Human perception
  - Discrete Cosine Transform(DCT)
- Non block-based schemes
  - Vector Quantization(VQ)
  - Sub-band

Data Transformation



Time Domain **Frequency Domain**

Discrete Cosine Transform

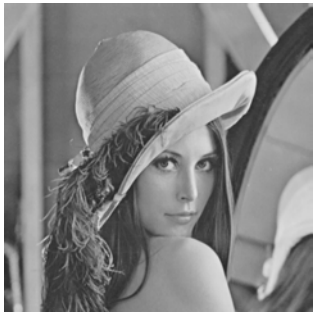
- Block size: 8 x 8
- Two-dimensional DCT:

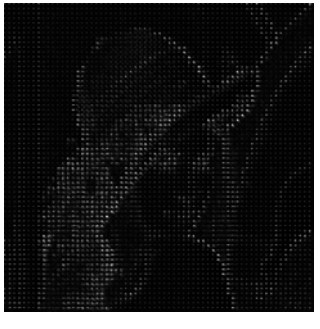
$$F(u,v) = \frac{2}{N} C(u)C(v) \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} f(x,y) \cos\frac{2\pi(2x+1)u}{4N} \cos\frac{2\pi(2y+1)v}{4N}$$

$$C(u), C(v) = \begin{cases} 1/\sqrt{2}, & u,v = 0 \\ 1, & \text{otherwise} \end{cases}$$

  - Separable -- row-column method
  - Most large coefficients concentrate on the upper-left corner
  - Quantization and zig-zag scan

Discrete Cosine Transform



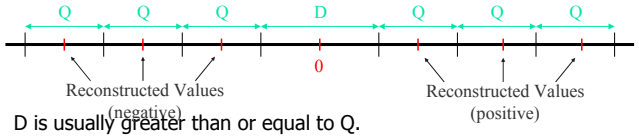


Quantization

- Usually the only **lossy** operation that removes perceptual irrelevancy.
- A **many-to-one mapping** that reduces the number of possible signal values at the cost of introducing errors.
- Often act a **control knob** for trading off image quality for bit rate.
- Uniform quantization versus optimized non-uniform quantization
- Uniform Quantizer with Dead-zone
- Can include the consideration of Human Visual System (HVS)

Uniform Quantizer with Dead-zone

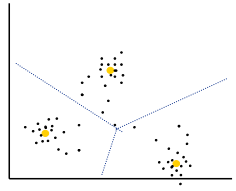
- Q: Quantizer step size D: Dead-zone size



- D is usually greater than or equal to Q.
  - Larger D increases the number of signal values quantized to 0.
  - The 0 index usually needs fewest bits to encode.
  - Thus adjusting D can get better rate-distortion optimization.
- When D=Q, the quantizer is degenerated to the normal uniform quantier.

Optimized Non-Uniform Quantizer

- **Design of the optimized quantizer:**
  - **Optimization Criterion:** *Expectation of Distortion*
    - Sometimes with a constraint of maximal bit rate (very complex)
  - Designed according to a mathematical model or real (training) data
- Designing quantizer from training data:
  - **Generalized Lloyd algorithm or LBG (Linde/Buzo/Gray) algorithm**

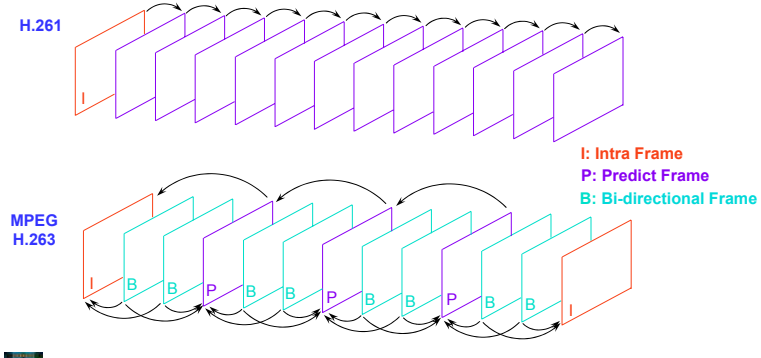


● : Reconstructed value point, usually the centroid

Temporal Redundancy

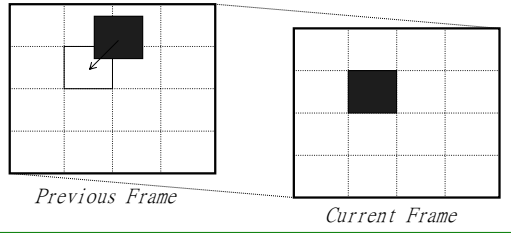
- Inter-frame coding(P-frame)
- Motion-compensated coding
  - Block-based motion
  - Object-based motion
  - Pel-based motion

Coding of Moving Pictures

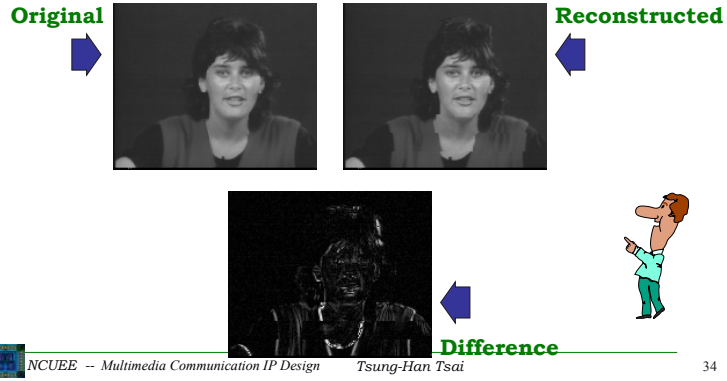


Motion Compensation

- Motion Estimation : Block matching
- Motion Compensation : Code prediction error



Motion Compensation/Estimation



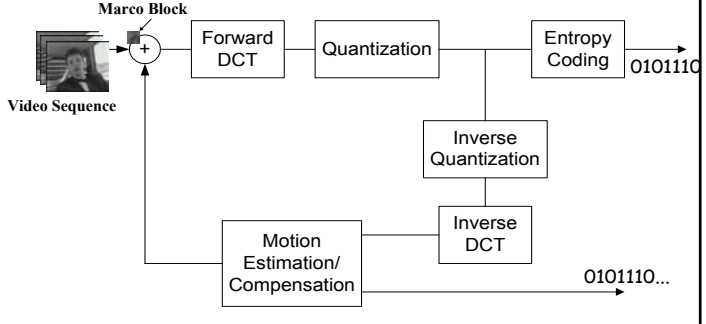
Variable Length Coding

- To reduce the statistical redundancy
- Probability-based codewords arrangement
- Huffman Coding

S1	0.45	1
S2	0.3	00
S3	0.15	010
S4	0.1	011

  - S4 S1 S2 S2 S3 S1 <---> 011100000101
- Arithmetic coding

Video Coder



METHODS AND STANDARDS FOR LOSSLESS COMPRESSION

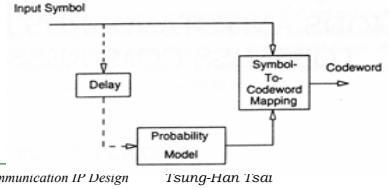
- Outline
  - Preliminaries
  - Huffman encoding & decoding
  - Arithmetic encoding & decoding
  - Standards for lossless compression

Introduction

- **Definition:** Compression methods for which the original uncompressed data set can be recovered exactly from the compressed stream
- **Usage:** digital medical data , bitonal image, artwork design, ...

Generic Model

- Symbol-to-codeword mapping function is usually referred to as entropy coding
- "Delay" is needed only when the the probability model is dynamically estimated



Partitioning

- Q1: How many symbols does the following string have?
  - A B C D E F （assume 8-bit ASCII format）
- Q2: Could a 12-bit precision symbol be fed into a 8-bit precision compression system?

Partitioning (con't)

- Answer to Q1:

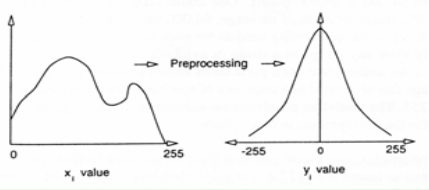
6 symbols (8-bit/sym) 、 3 symbols (16-bit/sym)
2 symbols (24-bit/sym) 、1 symbols (48-bit/sym)
48 symbols (1-bit/sym)

- Answer to Q2: That's OK!

(But some effect must taken into consideration,
ex: The degradation of compression ratio because of reduction of symbols)

Differential Coding

- Pixels in the image, x_i (i = 1~ N) Instead coding x_i directly, we can code $y_i = x_i - x_{i-1}$, because y_i have better probability distribution (y_i is called prediction residual of x_i)

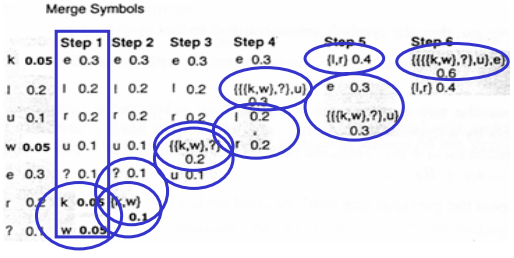


Huffman Encoding

- 1、information needed : prob. of symbols

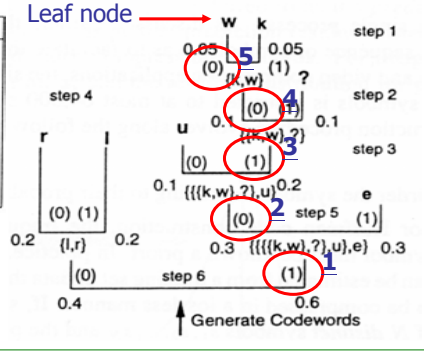
Symbol	Probability	Codeword
k	0.05	10101
l	0.2	01
u	0.1	100
w	0.05	10100
e	0.3	11
r	0.2	00
?	0.1	1011

- 2、Merge Symbols



Huffman Encoding (con't)

Symbol	Probability	Codeword
k	0.05	10101
l	0.2	01
u	0.1	100
w	0.05	10100
e	0.3	11
r	0.2	00
?	0.1	1011



- Binary tree ->
- l_{avg}= 2.6 = $\Sigma l_i p_i$

Properties of Huffman Codes

- **Entropy of source S** :

$$H(S) = \eta = \sum p_i \log_2 \frac{1}{p_i}$$

  - Average length boundary: $\eta \leq \bar{l}_{avg} < \eta + 1$
  - Take last slide for example: $2.546439 \leq 2.6 < 2.646439$
- No codeword is a prefix for another code word, so Huffman codes are uniquely decodable
- Because of it's variable length, if some bits miss, then all the data are lost (Inserting marker will improves this shortcoming)

Huffman Decoding

- Bit-Serial Decoding
  - Input: bit by bit with a fixed ra
  - Output: If a codeword is found corresponding symbol and disc

Symbol	Probability	Codeword
k	0.05	10101
l	0.2	01
u	0.1	100
w	0.05	10100
e	0.3	11
r	0.2	00
?	0.1	1011

input 1 0 1 0 1 → Symbol:K

input 0 1 → Symbol:l

input 1 1 → Symbol:e

- Constant input rate, variable output rate

Huffman Decoding(con't)

- Lookup-Table-Based Decoding
  - Buffer L (max codeword len
  - Every entry should be conta
    2^L entry lookup table is nee

Symbol	Probability	Codeword
k	0.05	10101
l	0.2	01
u	0.1	100
w	0.05	10100
e	0.3	11
r	0.2	00
?	0.1	1011

input, 5 bits (L=5)

1 0 1 0 1 → Symbol:K

discard

0 0 1 1 0 → Symbol:l

0 1 0 0 1 → Symbol:u

- Variable input rate, constant output rate

Huffman codes with constrained Length

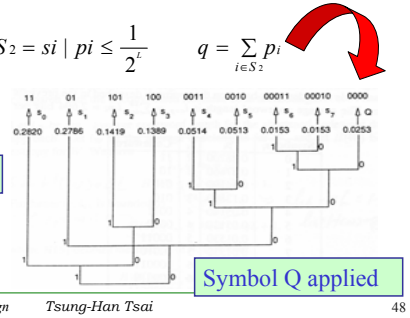
- Method 1:
  - Partition S into two sets S1 and S2 as

$$S_1 = si \mid pi > \frac{1}{2^L} \qquad S_2 = si \mid pi \leq \frac{1}{2^L} \qquad q = \sum_{i \in S_2} p_i$$

Assume L=7
1/128=0.0078125

Unconstrained, max 9 bits

8	0.00720	6	000011
9	0.00680	6	000010
10	0.00380	7	0000011
11	0.00320	7	0000010
12	0.00190	7	0000001
13	0.00130	8	00000001
14	0.00070	9	000000001
15	0.00040	9	000000000



Symbol Q applied

Huffman codes with constrained Length (con't)

- Constrained-length Huffman codewords

Symbol s_i	p_i	l_i	Codeword
0	0.28200	2	11
1	0.27860	2	10
2	0.14190	3	011
3	0.13890	3	010
4	0.05140	4	0011
5	0.05130	4	0010
6	0.01530	5	00011
7	0.01530	5	00010
8	0.00720	6	000011
9	0.00680	6	000010
10	0.00380	7	0000011
11	0.00320	7	0000010
12	0.00190	7	0000001
13	0.00130	8	00000001
14	0.00070	9	000000001
15	0.00040	9	000000000

Symbol i	p_i	l_i	Codeword	Additional
0	0.28200	2	11	
1	0.27860	2	01	
2	0.14190	3	101	
3	0.13890	3	100	
4	0.05140	4	0011	
5	0.05130	4	0010	
6	0.01530	5	00011	
7	0.01530	5	00010	
8	0.00720	11	0000	0001000
9	0.00680	11	0000	0001001
10	0.00380	11	0000	0001010
11	0.00320	11	0000	0001011
12	0.00190	11	0000	0001100
13	0.00130	11	0000	0001101
14	0.00070	11	0000	0001110
15	0.00040	11	0000	0001111

- Shortcoming: No guarantee for constant $\bar{N}_{avg}$ = 2.8057

Huffman codes with constrained Length (con't)

- Method 2: Ad-Hoc Design
  - For a max codeword length L, define a threshold $T=2^{-L}$
  - if pi ≤ T, set pi=T,then construct Huffman table

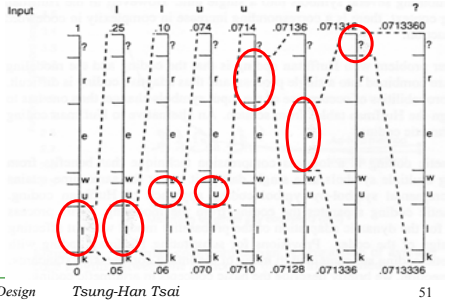
Symbol i	p_i	l	Codeword	Reordered l	Reordered Codeword
0	0.28200	2	11	2	11
1	0.27860	2	01	2	01
2	0.14190	3	101	3	101
3	0.13890	3	100	3	100
4	0.05140	4	0010	4	0010
5	0.05130	4	0001	4	0001
6	0.01530	6	001100	6	001100
7	0.01530	6	001101	6	001101
8	0.00720	7	0011110	6	000010
9	0.00680	7	0011111	6	000011
10	0.00380	7	0011100	6	000000
11	0.00320	7	0011101	6	000001
12	0.00190	6	000010	7	0011110
13	0.00130	6	000011	7	0011111
14	0.00070	6	000000	7	0011100
15	0.00040	6	000001	7	0011101
l_{avg}			2.7308		2.7141

rearrange

Arithmetic Coding

- Assume we want to code "*lluure?*"
  - Huffman:18 bits
  - Now: 16 bits
    - 0.0713348389
    $=2^{-4}+2^{-7}+2^{-10}$
    $+2^{-15}+2^{-16}$

Symbol	Probability	Codeword
k	0.05	10101
l	0.2	01
u	0.1	100
w	0.05	10100
e	0.3	11
r	0.2	00
?	0.1	1011

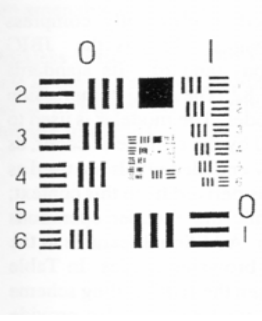


Implementation Issues

- Incremental output
  - Send a number which contain current symbol information, this is called incremental encoding
- High-precision arithmetic
  - Precision must be carefully considered to avoid overflow or underflow
- Probability modeling
  - Adaptive model: update along with input data

Standards for Lossless Compression

- Facsimile Compression Standards



Standards for Lossless Compression (con't)

- Lossless JPEG Standard
  - Prediction it! r=y-X (r: prediction residual)

	c	b
	a	X

a、b、c have already known to decoder

$$y = 0$$
$$y = a$$
$$y = b$$
$$y = c$$
$$y = a + b - c$$
$$y = a + \frac{b-c}{2}$$
$$y = b + \frac{a-c}{2}$$
$$y = \frac{a+b}{2}$$

Possible candidate

Standards for Lossless Compression (con't)

- Determine it's category, then encode (category ,value)
  - If r>0, value is it's binary value
  - If r<0, value is 1's complement of it's absolute value, with this way, if MSB of value is zero then the prediction residual is negative

Category	Prediction Residual
0	0
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
11	-2047, .., -1024, 1024, .., 2047
12	-4095, .., -2048, 2048, .., 4095
13	-8191, .., -4096, 4096, .., 8191
14	-16383, .., -8192, 8192, .., 16383
15	-32767, .., -16384, 16384, .., 32767
16	32768

Note:The bits used to represent the value equal to the category

Standards for Lossless Compression (con't)

- Let a=100 ,b=191 ,c=100
- X=180, y=(a+b)/2=145
- r=145-180= -35, which belongs to category 6
- The binary number for -35 is 011100, and the Huffman code for 6 is 1110. Thus the overall code is 1110011100 (note: 35=100011, it's one's complement = 011100)

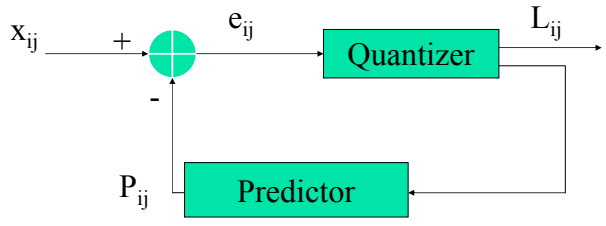
	c	b
	a	X

Category	Prediction Residual
0	0
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
11	-2047, .., -1024, 1024, .., 2047
12	-4095, .., -2048, 2048, .., 4095
13	-8191, .., -4096, 4096, .., 8191
14	-16383, .., -8192, 8192, .., 16383
15	-32767, .., -16384, 16384, .., 32767
16	32768

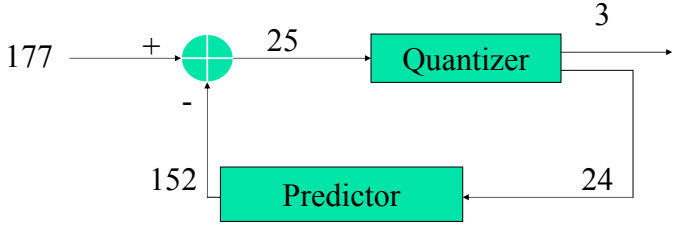
Fundamentals of Lossy Video Compression

- Preliminaries
- Sample-based Coding
- Block-based Coding
- DCT and IDCT
- Motion Estimation and Compensation
- Advanced Algorithms and Architectures

Sample-based Coding



Sample-Based Coding Example



Block-based Coding

- Data highly relates in distance less than 8
- Therefore a 8x8 block is used
- Processing in frequency domain
  - The low frequency component is more dominant than the high frequency ones.
  - Eyes are insensible to high-frequency components

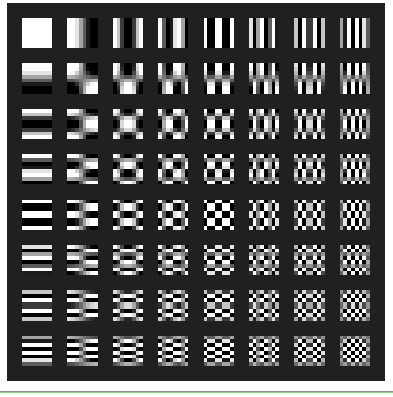
Block-based Coding (continued)

- Symmetric transform
  - 2D transformation can be separated into two 1D transformation
  - Let X:source image, Tc, Tr: column and row transformation
    then $Y=Tc\ X\ Tr^T$
  - If the transformation is symmetric,
    $Y = T\ X\ T^T$

DCT - Discrete Cosine Transform

- Original image can be represented by many different methods, which use different bases
- For a basis β, the image can be expressed as
  - $X_{00}\ \beta_{00}+X_{01}\ \beta_{02}+ \ldots + X_{77}\beta_{77}$

Block basis of the DCT



1-D DCT

- Pixel basis
  - $e_0, e_1, e_2, \ldots, e_7$
- DCT basis

$$\beta_k = \begin{pmatrix} 0.5c(k)\cos(k\pi/16) \\ 0.5c(k)\cos(3k\pi/16) \\ 0.5c(k)\cos(5k\pi/16) \\ 0.5c(k)\cos(7k\pi/16) \\ 0.5c(k)\cos(9k\pi/16) \\ 0.5c(k)\cos(11k\pi/16) \\ 0.5c(k)\cos(13k\pi/16) \\ 0.5c(k)\cos(15k\pi/16) \end{pmatrix}$$

- By orthogonal projection
  - $x_i = \Sigma_k \langle x_i, \beta_k \rangle \beta_k$, k = 0..7
  - x_i means the i-th row of the block
  - $\langle x_i, \beta_k \rangle = \Sigma_j x_{ij} \beta_{jk}$
    $= 0.5c(k) \Sigma_j x_{ij} \cos((2j + 1)k\pi/16)$
    $= z_{ik}$

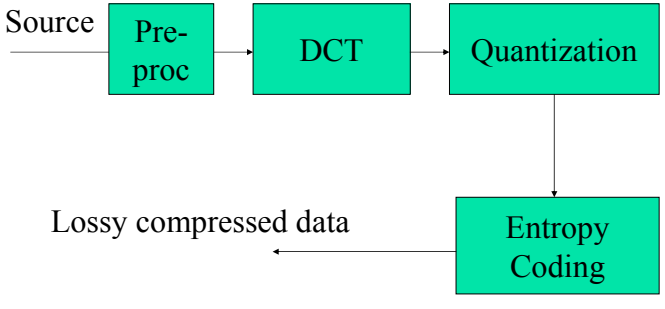
2-D DCT

- Cascaded by two 1-D DCT
- Row and column operation, respectively
  - $Y = TXT^T$
- Note: c(k) $= 1/\sqrt{2}$ for k = 0
  $= 1$ otherwise
- 2-D DCT in eq. 3.20

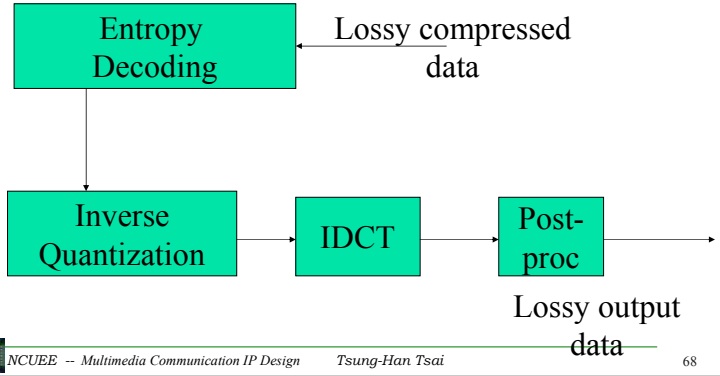
1-D I-DCT

- Inverse DCT
- Similar to 1-D DCT
- DCT: Pixel basis => DCT basis
- IDCT: DCT basis => Pixel basis
- 2-D I-DCT can be cascaded by two 1-D I-DCT
  - eq. 3.22

DCT-based Image Coding



DCT-based Decoding



Quantization/I-Quantization

- Eyes are insensible to high-frequency components
- The greater quantizer means greater loss
- Lower frequency component has smaller quantizer
- High frequency component has greater quantizer
- The quantization tables in the encoder and decoder are the same

Pre-processing and Post-processing

- Goal: make a zero average
- For YCbCr, Y has average of 128 while Cb and Cr have average of zero
- Pre-processing: Y' = Y - 128
- Post-processing: Y = Y' + 128

DCT Flowgraph

- 2D DCT can be achieved by two 1D DCT
  - Row-transformation follows the column-transformation
  - 1D DCT is a summation operation (eq 3.19)
- 1D DCT
  - Every pair in all stage is in the form

$$p[i] = t[j]\, b + t[i]\, a$$
$$p[j] = t[j]\, b - t[i]\, a$$

Fast Scaled DCT

- Goal: reduce multiplications in DCT
- Involve
  - Coefficients of last stage DCT
  - Quantization matrix Q
- The multiplication in the last stage of DCT can be combined with quantizers before DCT starts

Fast Scaled DCT(cont'd)

- Column DCT => Row DCT => Q
- Q: z[i] = y[i]/q[i]
- At last stage
  - y[i] = t[j] b + t[i] a = a (t[j] + c t[i])
    y[j] = t[j] b - t[i] a = a (t[j] – c t[i])
  - Since later the y[] has to be quantized, we modify the quantizer such that
    q'[i] = q[i] * a
    Thus the last stage can be modified
  - y[i] = t[j] + c t[i]
    y[j] = t[j] – c t[i]

Multiply-Accumulate DCT

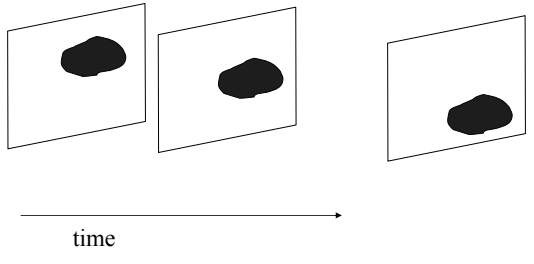
- Every stage in DCT is equivalent to a multiply-accumulate operation
- Because all operations is arranged to multiply-accumulation, the design in hardware can be regulated by MACs

Multiply-free DCT

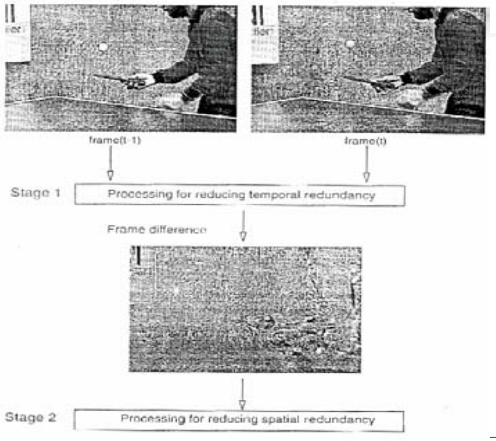
- Goal: use integer adds and shifts to replace multiplications
- y[i] = t[j] b + t[i] a
  y[j] = t[j] b - t[i] a
  Find integer of a and b
- The multiple error can be fixed by the scaled quantizers after the last stage
- Make all the 1D DCT transformation matrix coefficients be integers (page 83 and 84)
- Premise
  - Coefficients obey the magnitude relations in the original matrix: a >= b >= c >= d
  - Preserve orthogonality: ab=ac + bd + cd

Motion Estimation and Compensation

Image Sequence Model



Two-stage video coding process



Stage 1 : Reducing Temporal Redundancy

- Segment a frame into macroblocks.
- Output energy is increased with the degree of temporal redundancy.
- Interframe coder .

Stage 2 : Reducing Spatial Redundancy

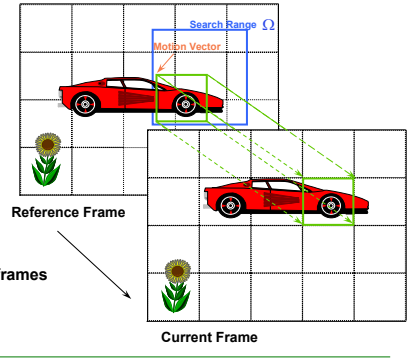
- Processing the difference frame (spatially correlated) from stage 1.
- Using DCT coding.
- Intraframe coder.
- Hybrid coding method.

Block-Matching Motion Estimation

$X_t(p,q)$

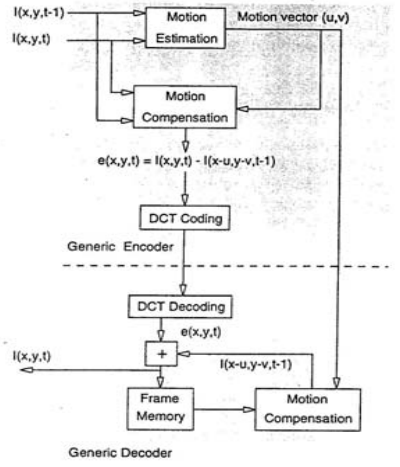
the block at location (p,q) in t-th frame

Motion Vector
$V_t(p.q) = (Veci, Vecj)$
the location in the search range Ω
that has the maximum correlation value
between blocks in temporally adjacent frames



Motion Compensation

- The process of compensating for the displacement of moving objects from one frame to another.
- It is preceded by motion estimation.
- Similar with DPCM.



$$k = \sqrt{u^2 + v^2}$$

Figure 4.3 Generic hybrid video coder and decoder.

Performance Evaluation

Questions before using interframe coding:

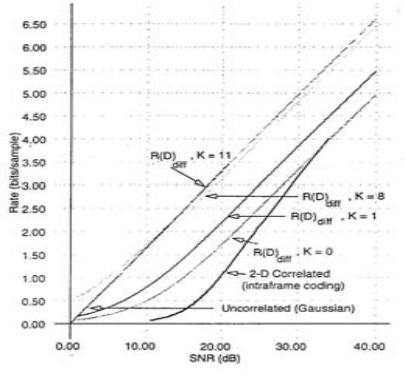
- How does intraframe coding compare to interframe coding?
- How do hybrid coding schemes compare to intraframe or interframe coding alone?
- Does the added complexity of motion compensation warrant its use

- 2-D correlated (intraframe)
- Differencing (interframe)
- Motion-compensation (interframe)
- Hybrid (interframe /intraframe)

Frame differencing with no motion compensation

- K is the displacement distance
- K>8 offers no improvement, means no correlation
- All cases show that FD is good enough. Intraframe coding does the job best .

$$k = \sqrt{u^2 + v^2}$$



Motion-compensated prediction

- There are 40% improvement even at high motion-estimation error of +/- 1 pixel.
- Different motion precision achieve different performance.

$$R = \frac{1}{2}\log_2 \frac{\sigma_e^2}{D}$$

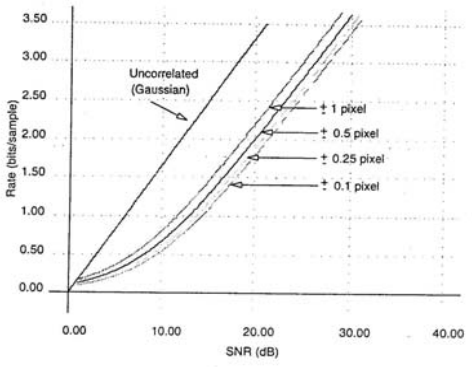


Figure 4.6 Rate-distortion performance for motion-compensated prediction with no intraframe coding.

Motion-compensated prediction v.s. intraframe coding

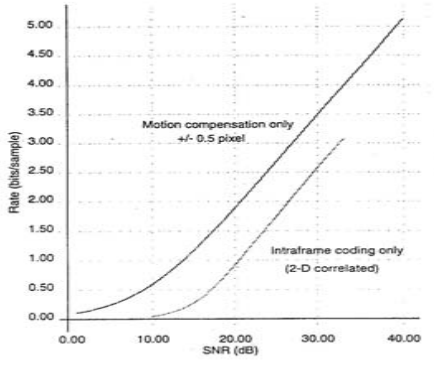


Figure 4.7 Rate-distortion performance for motion-compensated prediction versus intraframe coding.

Hybrid Coding: Motion-compensated video followed by intraframe coding

- Hybrid coding can get better results.

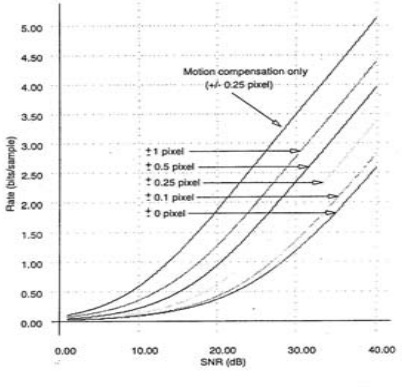
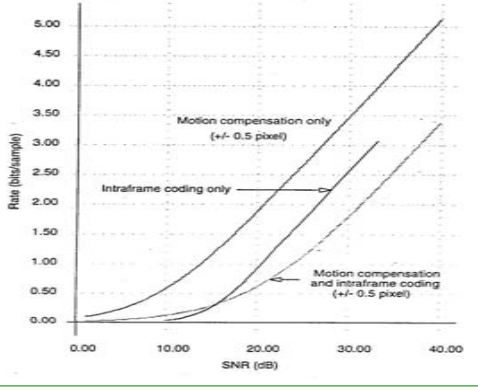


Figure 4.8 Rate-distortion functions for hybrid video coding.

Rate-distortion functions for Hybrid, interframe, and intraframe video coding



R(D) Summary

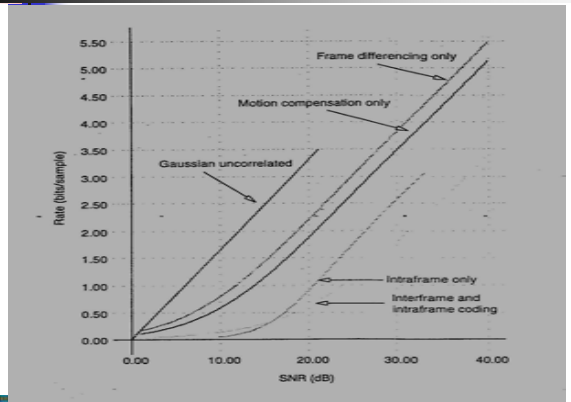


Figure 4.10 Rate-distortion functions for various video coding schemes.

Motion-compensated prediction

- Hybrid coding method is quite effective.
- In interframe coding , the motion-estimation problem takes an important role.

Motion Compensation Algorithms in Video Coding

Motion Estimation

- ***Objective***
  - ***Predict current frame from neighboring frames***
- ***Motion Estimation Algorithm***
  - ***Pixel-based method (Pel-Recursive Algorithm)***
    - ***Large computation overhead***
  - ***Block-based method***
    - ***Regularity and simplicity***
    - ***Suitable for hardware implementation***
  - ***Object-based method (content-based)***
- ***Crucial to make possible a high degree of video compression***

Block-Matching Motion Estimation

$X_t(p,q)$

the block at location (p,q) in t-th frame

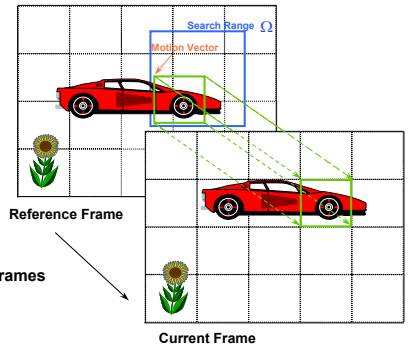
Motion Vector

$V_t(p.q) = (Veci, Vecj)$

the location in the search range Ω

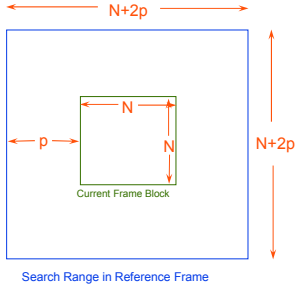
that has the maximum correlation value

between blocks in temporally adjacent frames



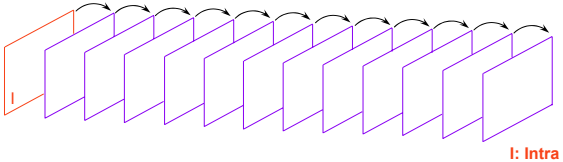
Factors of Affecting Block-Matching Algorithm(BMA)

- ***Search Algorithm***
- ***Match Criterion***
- ***Search Range***



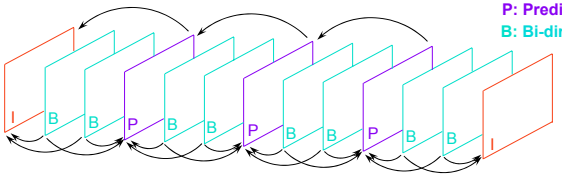
Coding of Moving Pictures

H.261



I: Intra Frame
P: Predict Frame
B: Bi-directional Frame

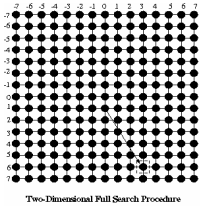
MPEG
H.263



Brief Review of Previous Algorithms

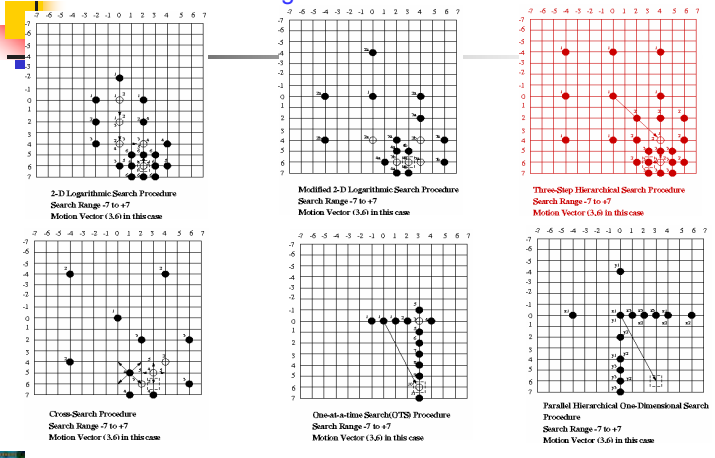
Search Algorithms

- ***Two-Dimensional Full Search(2DFS)***
  - ***Exhaustive search***
  - ***Extremely Large Computation***
- ***Fast Search***
  - ***Assumption: Monotonic Distortion Function***
  - ***Reduce computation at the expense of accuracy***
  - ***2-D Logarithmic Search***
  - ***Modified 2-D Logarithmic Search***
  - ***Three-Step Hierarchical Search***
  - ***Cross Search***
  - ***One-at-a-time Search***
  - ***Parallel Hierarchical One-Dimensional Search***
  - ***One-Dimensional Full Search***



Two-Dimensional Full Search Procedure
Search Range -7 to +7
Motion Vector (3,6) in this case

Fast Search Algorithms



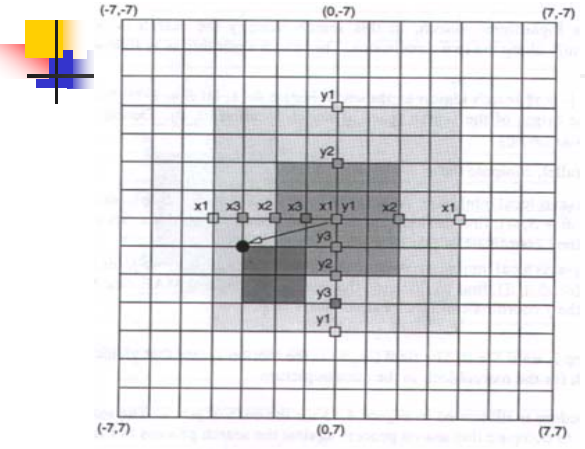


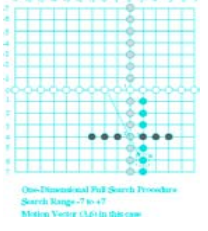
Figure 4.13 Example of PHODS strategy.

Fast Search Algorithms

One-Dimensional Full Search

For (i= -P to +P) Veci'= i | min D(i,j)
For (j= -P to +P) Vecj'= j | min D(Veci',j)
For (i= -P/2 to +P/2) Veci = i | min D(i+Veci',Vecj')
For (j= -P/2 to +P/2) Vecj = j | min D(Veci,j+Vecj')
Motion Vector = (Veci,Vecj)

- *Hardware-Oriented*
- *Full Data Reuse*
- *Regular Data Flow*
- *Fixed-Step Operation*
- *Good Performance*



Ref: Mei-Juan Chen, Liang-Gee Chen and Tzi-Dar Chiueh, "One-Dimensional Full Search Motion Estimation Algorithm for Video Coding", IEEE Transactions on Circuits and Systems for Video Technology, Vol.4, No.5, October 1994.

Performance Comparison

	Miss America	Train Calendar	Table Tennis
PSNR (dB)			
2DFS	38.64	19.57	25.48
1DFS	38.39	19.45	24.55
3Step	38.20	19.40	24.06
Entropy (bits/pixel)			
2D FS	3.49	5.93	5.27
1DFS	3.52	5.94	5.39
3Step	3.56	5.94	5.47

30 frames/sec Sequences

Matching Criteria

- *Cross-Correlation Function(CCF)*
- *Mean Squared Error(MSE)*
- *Mean Absolute Error(MAE)*
- *Pel Difference Classification(PDC)*
- *Minimized Maximum Error (MiniMax)*

$x_t(k,l)$: luminance for the location (k,l) in $X_t(p,q)$

$x_{t-1}(k+i,l+j)$: luminance for the shifted location by *i* pels and *j* lines within the search range

Matching Criteria

Cross-Correlation Function(CCF)

$$CCF(i,j)=\frac{\sum_{k=1}^{N}\sum_{l=1}^{N} x_t(k,l)\cdot x_{t-1}(k+i,l+j)}{\left[\sum_{k=1}^{N}\sum_{l=1}^{N} x_t^2(k,l)\right]^{\frac{1}{2}}\left[\sum_{k=1}^{N}\sum_{l=1}^{N} x_{t-1}^2(k+i,l+j)\right]^{\frac{1}{2}}}$$

(Veci,Vecj)=(i,j) | max *CCF(i,j)*

Mean Squared Error(MSE) --> most widely used in software

$$MSE(i,j)=\frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N}\left[x_t(k,l)-x_{t-1}(k+i,l+j)\right]^2$$

(Veci,Vecj)=(i,j) | min *MSE(i,j)*

Matching Criteria

Mean Absolute Error(MAE) ➡ ***Most Widely Used in hardware***

$$MAE(i,j)=\frac{1}{N^2}\sum_{k=1}^{N}\sum_{l=1}^{N}\left|x_t(k,l)-x_{t-1}(k+i,l+j)\right|$$

(Veci,Vecj)=(i,j) | min *MAE(i,j)*

Pel Difference Classification(PDC)

$$T(k,l,i,j)=\begin{cases}1, & \left|x_t(k,l)-x_{t-1}(k+i,l+j)\right|\le Threshold\\ 0, & otherwise\end{cases}$$

$$G(i,j)=\sum_{k=1}^{N}\sum_{l=1}^{N} T(k,l,i,j)$$

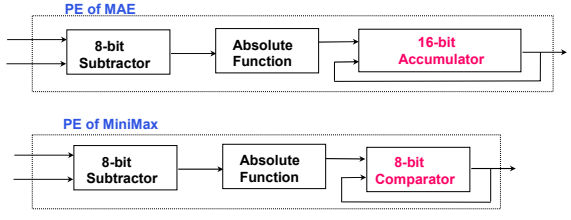
(Veci,Vecj)=(i,j) | max *G(i,j)*

Matching Criteria

Minimized Maximum Error (MiniMax) ➡ Hardware Reduction

$G(i,j)$= max $\left|x_t(k,l)-x_{t-1}(k+i,l+j)\right|$

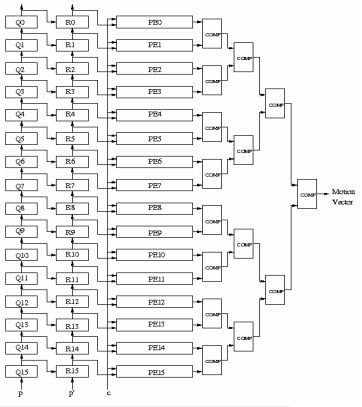
(Veci,Vecj)= (i,j) | min *G(i,j)*



Architecture Design and Chip Implementation

- One-Dimensional Full Search
  - MiniMax Criterion

Technology	0.8 um CMOS
Chip Size	6.5 mm x 5.3 mm
Number of Transistors	48170
Chip Clock	43 MHz
Number of Pads	61
Power Dissipation	383 mW



Subjective Quality

1083

Table Tennis I

Min-max

MAE

Table Tennis II

Min-max

MAE

Salesman

Min-max

MAE

Train & Calendar

Min-max

MAE

Other Approaches: Pixel Subsampling for MAE calculations

1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4
1	2	1	2	1	2	1	2
3	4	3	4	3	4	3	4

Figure 4.15 Pixel decimation for block matching in an 8 × 8 block.

Pixel Subsampling for MAE calculations

- For each MAE value , we use only 1/4 of the pixels.
- But every pixel in the block will be used.
- It minimizes the possibility of not considering one-pixel-wide horizontal , vertical and diagonal lines .

Projections for MAE calculations

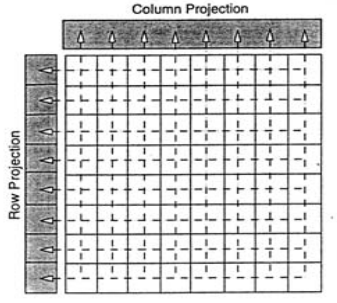


Figure 4.16 Row and column projection of pixels in an 8 × 8 block.

Predictive Motion Estimation Using Boundary/Side Match

- ***Motivation***
  - ***Relieve huge computation complexity imposed on full search for large search range motion estimation***
  - ***MPEG and HDTV applications***
  - ***Little attention paid in the past***
- ***Goal***
  - ***Provide solutions for the computation reduction of increasing search area***
  - ***Object-based concept***

Predictive Motion Estimation Algorithms

$$V_t(p,q) = \hat{E}[\hat{V}_t(p,q)] + \delta V_t(p,q) \quad , \quad |\delta V_t(p,q)| \le \omega$$

$\hat{E}[\hat{V}_t(p,q)]$:Predicted Motion Vector

- ***Temporal Prediction***
  - ***Inter-Frame Prediction***
- ***Spatial Prediction***
  - ***Inter-Block Prediction***
  - ***Median Vector Prediction***
  - ***Proposed Boundary Match Prediction***
  - ***Proposed Side Match Prediction***

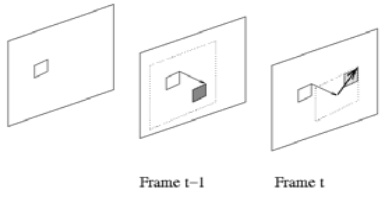
$\delta V_t(p,q)$:Refined Displacement

ω :Reduced Search Range

(e.g. one-quarter of search range Ω

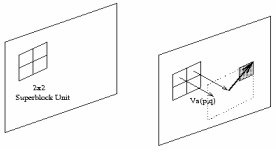
Inter-Frame Prediction

$$E[\hat{V}_t(p,q)] = V_{t-1}(p,q)$$



Inter-Block Prediction

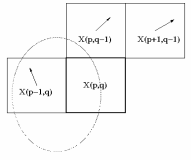
$$E[\hat{V}_t(p,q)] = V_s(p,q)$$



Median Vector Prediction

$$E[\hat{V}_t(p,q)] = \arg \underset{v \in V_{cs}}{Median}\ V$$

$$V_{cs} = \{V_t(p,q-1), V_t(p-1,q), V_t(p+1,q-1)\}$$



Boundary/Side Match

- *A motion object often covers many small blocks*
- *Blocks in the same object have similar motion vectors*
- *Spatial Neighboring Blocks*
  - *Very likely move in the same direction with similar velocities*
  - *Highly correlated or dependent*
- *Utilize the high spatial correlation between the boundary pixels of adjacent blocks to determine a more accurate initial motion estimation center*
- *Other applications*
  - *Vector quantization (VQ)*
  - *Recovery of lost vector or channel error*
  - *Transform coded image reconstruction*

Boundary Match

$$[X(p,q)]_J^c = (x_{J1}, x_{J2}, \cdots, x_{Jn})^T \quad \text{column vector}$$

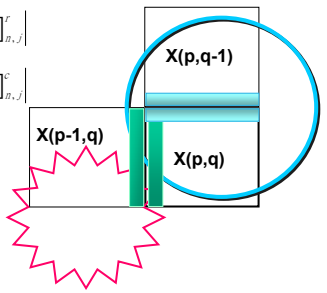
$$[X(p,q)]_J^r = (x_{1J}, x_{2J}, \cdots, x_{nJ}) \quad \text{row vector}$$

$$d_{bm,V_t(p,q-1)} = \sum_{j=1}^{n} \left| [X(p,q)]_{1,j}^r - [X(p,q-1)]_{n,j}^r \right|$$

$$d_{bm,V_t(p-1,q)} = \sum_{j=1}^{n} \left| [X(p,q)]_{1,j}^c - [X(p-1,q)]_{n,j}^c \right|$$

$$E[\tilde{V}_t(p,q)] = \arg \underset{v \in V_{cs}}{Min}\ d_{bm,v}$$

$$V_{cs} = \{V_t(p,q-1), V_t(p-1,q)\}$$



Side Match

$$d_{smU} = \sum_{j=1}^{n} \left| [X_{MC}(p,q)]_{1,j}^r - [X(p,q-1)]_{n,j}^r \right|$$

$$d_{smL} = \sum_{j=1}^{n} \left| [X_{MC}(p,q)]_{1,j}^c - [X(p-1,q)]_{n,j}^c \right|$$

$$d_{smR} = \sum_{j=1}^{n} \left| [X_{MC}(p,q)]_{n,j}^c - [X(p+1,q)]_{1,j}^c \right|$$

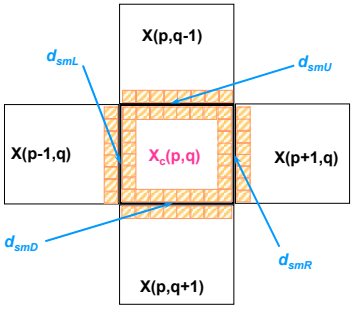
$$d_{smD} = \sum_{j=1}^{n} \left| [X_{MC}(p,q)]_{n,j}^r - [X(p,q+1)]_{1,j}^r \right|$$

$$d_{sm} = d_{smU} + d_{smL} + d_{smR} + d_{smD}$$

$$E[\tilde{V}_t(p,q)] = \arg \underset{v \in V_{cs}}{Min}\ d_{sm,v}$$

$$V_{cs} = \{V_t(p,q-1), V_t(p-1,q), V_t(p+1,q-1)\}$$

minimum gray-level transition across four boundaries



Computation Comparison

Algorithms	Search Range	Computation Complexity (SD/block)
Full Search	-p ~ p	$(2p+1)^2$ x m x n
Inter- Frame	-p/2 ~ p/2	$(p+1)^2$ x m x n
Inter- Block	-p ~ p - p/2 ~ p/2	[$1/4(2p+1)^2 + 3/4(p+1)^2$] x m x n
Median	- p/2 ~ p/2	$(p+1)^2$ x m x n +6
Boundary Match	- p/2 ~ p/2	$(p+1)^2$ x m x n + m + n
Side Match	- p/2 ~ p/2	$(p+1)^2$ x m x n+ 6(m + n)

Objective Performance Measurement

PSNR: Peak-to-Peak Signal-to-Noise Ratio

predicted image compared to the original image

$$PSNR = 10 \times \log_{10} \frac{255^2}{MSE} \quad \textbf{(dB)}$$

Entropy of Prediction Error (bits/pixel)

$$Entropy = -\sum_i p(e_i) \cdot \log_2 p(e_i) = \sum_i p(e_i) \cdot \log_2 \frac{1}{p(e_i)}$$

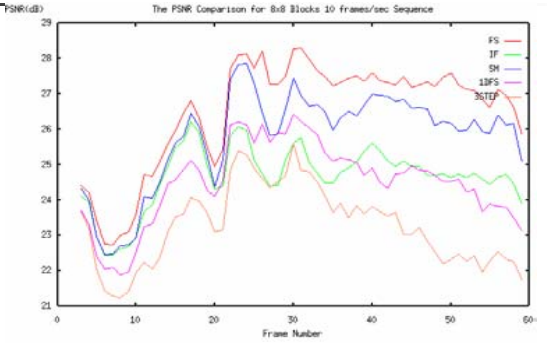
$p(e_i)$ **: probability of occurrence of prediction error pattern**

PSNR Comparison

Algorithms	30 frames/sec	15 frames/sec	10 frames/sec
Full Search	27.33	26.90	26.49
Inter-Frame	25.73	25.23	24.66
Inter-Block	26.09	25.52	25.22
Median	25.57	24.93	24.23
Boundary Match	26.40	26.03	25.68
Side Match	26.52	26.17	25.77
IDFS	26.23	25.09	24.49
3-Step	24.94	23.06	23.26

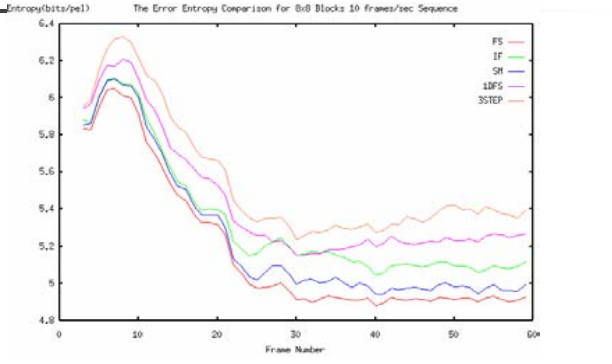
Block size 8x8 60 frames Table Tennis Sequence

PSNR vs Frame Number



block size 8x8 10 frames/sec Table Tennis Sequence

Error Entropy vs Frame Number



block size 8x8 10 frames/sec Table Tennis Sequence

Motion Vector Characteristics

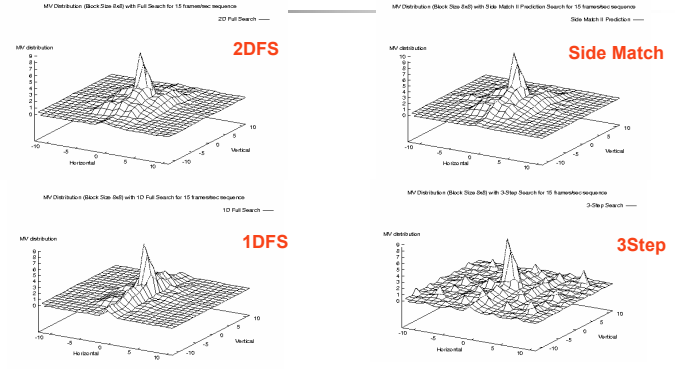
MV Euclidean Distance(ED) $(V_x-V_{x,fs})^2+(V_y-V_{y,fs})^2$

Hit Ratio(HR) Criterion: $|V_x-V_{x,fs}|<=2$ and $|V_y-V_{y,fs}|<=2$

Frame Rate	30 frames/sec		15 frames/sec		10 frames/sec	
Algorithms	ED	HR	ED	HR	ED	HR
Inter-Frame	23.52	72.92%	60.05	67.50%	74.67	62.46%
Inter-Block	13.05	76.88%	54.54	71.37%	72.82	67.55%
Median	20.63	73.33%	57.15	66.98%	82.41	60.64%
Boundary Match	10.08	79.70%	47.88	74.76%	66.91	70.73%
Side Match	9.97	80.93%	46.08	75.72%	65.18	71.86%
IDFS	26.71	61.97%	95.13	41.74%	132.27	31.55%
3-Step	23.14	50.81%	100.92	30.39%	144.47	23.80%

Block size 8x8 60 frames Table Tennis Sequence

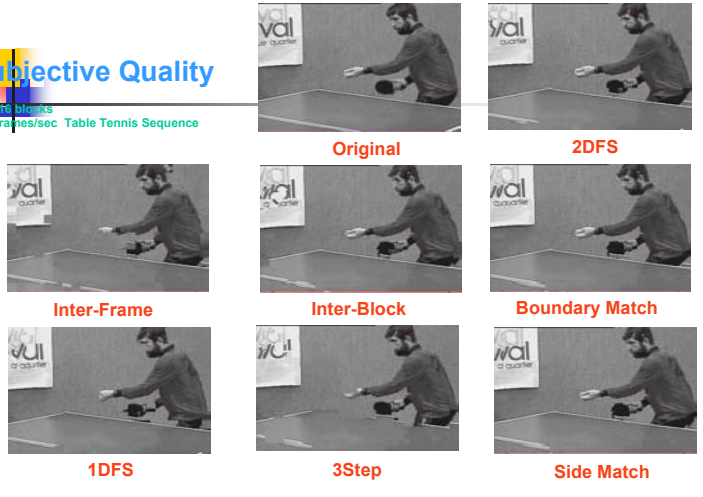
Motion Vector Distribution



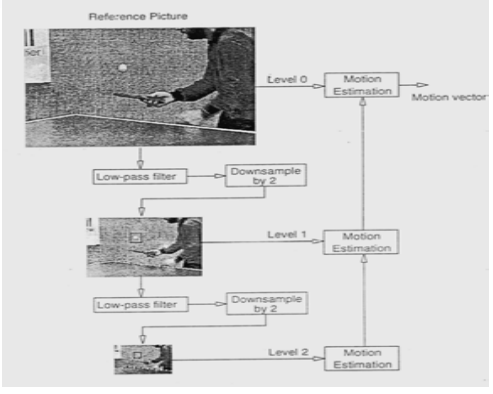
block size 8x8 15 frames/sec Table Tennis Sequence

Subjective Quality

16 by 16 blocks
15 frames/sec Table Tennis Sequence



Hierarchical Motion Estimation



Hierarchical Motion Estimation

1. **Level 2**
   - Picture size = 180 × 120, and macroblock size = 4 × 4.
   - Number of macroblocks = $\frac{Picture\ size}{Macroblock\ size}$ = 1,350. At 30 fps, we have 40,500 macroblocks.
   - Search parameter = $\lceil \frac{p}{4} \rceil = 4$.
   - Number of search locations = $(2 \times 4 + 1)^2 = 81$.
   - Number of operations per search location = macroblock size × 3 = 48.

   Complexity for Level 2 = 40,500 × 81 × 48 = 157.46 MOPS.

Hierarchical Motion Estimation

2. **Level 1**
   - Picture size = 360 × 240, and macroblock size = 8 × 8.
   - Number of macroblocks = 1,350. At 30 fps, we have 40,500 macroblocks.
   - Search parameter = 1.
   - Number of search locations = 9.
   - Number of operations per search location = macroblock size × 3 = 192.

   Complexity for Level 1 = 40,500 × 9 × 192 = 69.98 MOPS.

Hierarchical Motion Estimation

3. **Level 0**
   - Picture size = 720 × 480, and macroblock size = 16 × 16.
   - Number of macroblocks = 1,350. At 30 fps, we have 40,500 macroblocks.
   - Search parameter = 1.
   - Number of search locations = 9.
   - Number of operations per search location = macroblock size × 3 = 768.

   Complexity for Level 0 = 40,500 × 72 × 48 = 279.9 MOPS.

Hierarchical Motion Estimation

- It requires increased storage to keep pictures at different resolutions.
- Motion vector may be inaccurate for regions containing small objects .
- Low-pass filter can reduce noise.

Operations Comparison of Motion-Estimation Methods

Search Method	Operations per Macroblock	Operations for pictures 720 x 40 at 30 fps	
		$p = 15$	$p = 7$
Full-search	$(2p+1)^2 NM3$	29.89 GOPS	6.99 GOPS
Logarithmic	$(8\lceil log_2 p \rceil + 1)NM3$	1.02 GOPS	777.60 MOPS
PHODS	$(4\lceil log_2 p \rceil + 1)NM3$	528.76 MOPS	404.35 MOPS
Hierarchical	$[(2\lceil \frac{p}{4} \rceil + 1)^2 + 180] \frac{NM}{16} 3$	507.38 MOPS	398.52 MOPS

Table 4.1 Computational complexity and MOPS requirements for various motion-estimation algorithms using the MAE criterion and a [-p, p] search range.

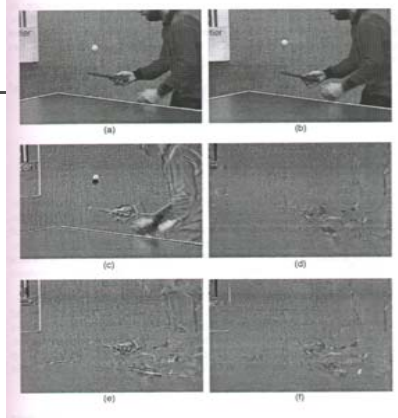


Figure 4.18 Performance comparison of motion-estimation methods. (a) Reference picture; (b) current picture; (c) simple frame differencing; (d) full-search; (e) logarithmic search; (f) three-level hierarchical search.

Sub-pixel-accurate motion estimation

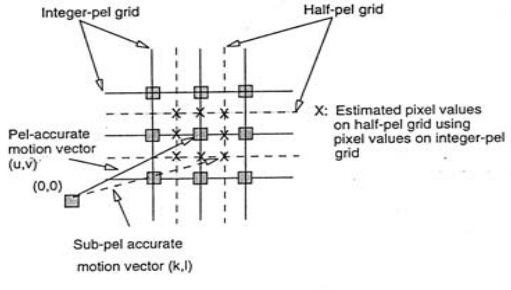
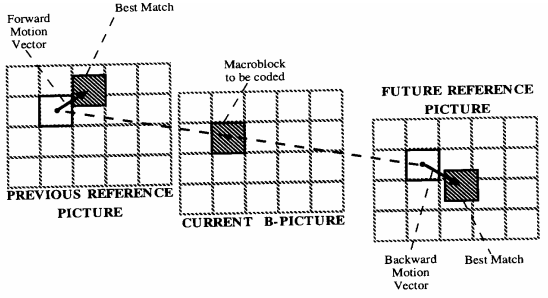


Figure 4.19 Half-pel accurate motion vector estimation.

Bidirectional Temporal Prediction for Progressive Video

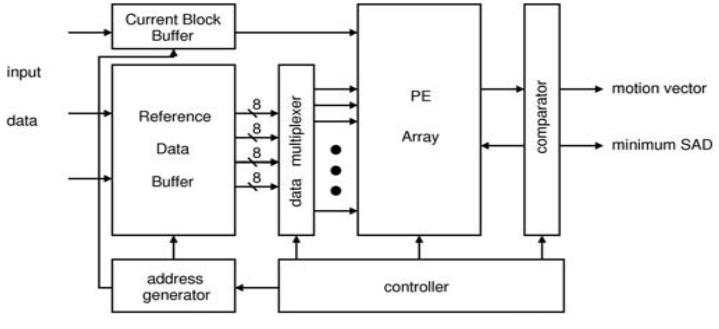
- Reference frame must be I or P-frame
- B-picture
- 2 MV



Motivation

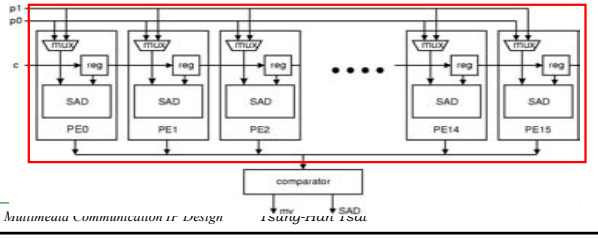
- various specifications of applications
  - video conference: 97.3M
    - QCIF, 15 fps, search range [-8,7]
  - MPEG-4 Core Profile Level 1 : 389.3M
    - QCIF, Max rate 5940 MB/s, search range [-8,7]
  - MPEG-4 Core Profile Level 2 : 6.229 G
    - CIF, Max rate 23760 MB/s, search range [-16,15]
  - scalable architecture
- data-dominated application

Motion Estimation Core



Module of Processing Element

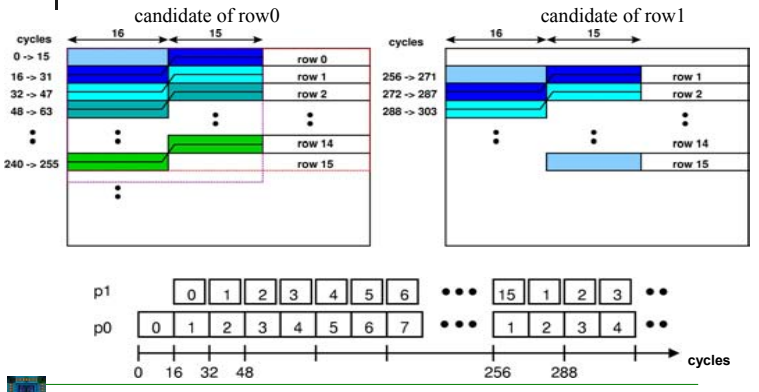
- one dimensional systolic array
  - M.T. Sun
- every PE is responsible for a candidate of motion vector



Data Flow of One Module

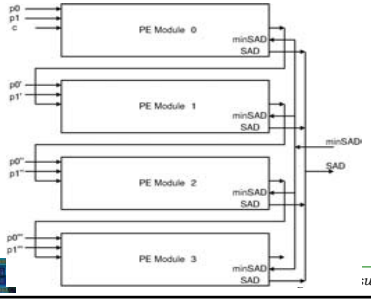
clocks	reference data		PE0	PE1	PE2	PE14	PE15
0	b(0,0)		a(0,0)-b(0,0)				
1	b(0,1)		a(0,1)-b(0,1)	a(0,0)-b(0,1)			
2	b(0,2)		a(0,2)-b(0,2)	a(0,1)-b(0,2)	a(0,0)-b(0,2)		
				.....			
14	b(0,14)		a(0,14)-b(0,14)	a(0,13)-b(0,14)	a(0,12)-b(0,14)	a(0,0)-b(0,14)	
15	b(0,15)		a(0,15)-b(0,15)	a(0,14)-b(0,15)	a(0,13)-b(0,15)	a(0,1)-b(0,15)	a(0,0)-b(0,15)
16	b(1,0)	b(0,16)	a(1,0)-b(1,0)	a(0,15)-b(0,16)	a(0,14)-b(0,16)	a(0,2)-b(0,16)	a(0,1)-b(0,16)
17	b(1,1)	b(0,17)	a(1,1)-b(1,1)	a(1,0)-b(1,1)	a(0,15)-b(0,17)	a(0,3)-b(0,17)	a(0,2)-b(0,17)
				.....			
29	b(1,13)	b(0,29)	a(1,13)-b(1,13)	a(1,12)-b(1,13)	a(1,11)-b(1,13)	a(0,15)-b(0,29)	a(0,14)-b(0,29)
30	b(1,14)	b(0,30)	a(1,14)-b(1,14)	a(1,13)-b(1,14)	a(1,12)-b(1,14)	a(1,0)-b(1,14)	a(0,15)-b(0,30)
31	b(1,15)		a(1,15)-b(1,15)	a(1,14)-b(1,15)	a(1,13)-b(1,15)	a(1,1)-b(1,15)	a(1,0)-b(1,15)
32	b(2,0)	b(1,16)	a(2,0)-b(2,0)	a(1,15)-b(1,16)	a(1,14)-b(1,16)	a(1,2)-b(1,16)	a(1,1)-b(1,16)
				.....			
256	b(1,0)	b(15,16)	a(0,0)-b(1,0)	a(15,15)-b(15,16)	a(15,14)-b(15,16)	a(15,2)-b(15,16)	a(15,1)-b(15,16)
257	b(1,1)	b(15,17)	a(0,1)-b(1,1)	a(0,0)-b(15,17)	a(15,15)-b(15,17)	a(15,3)-b(15,17)	a(15,2)-b(15,17)

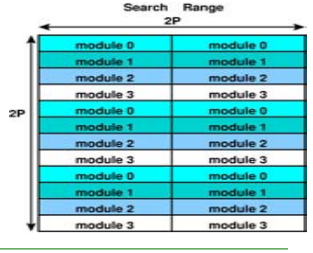
Reference Data Access



Example of Multiple Modules

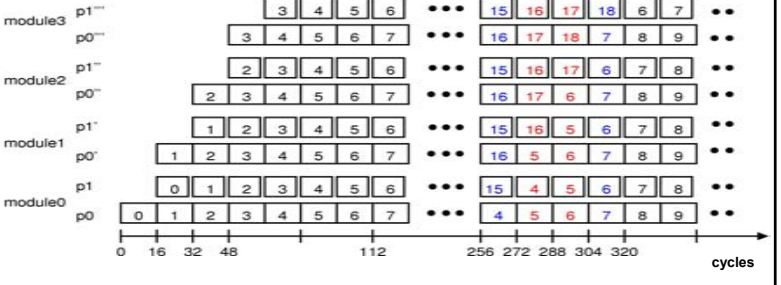
- cascade four modules
  - current block data propagate
  - reference data independent





Reference Data Access of Multiple Modules

- four modules

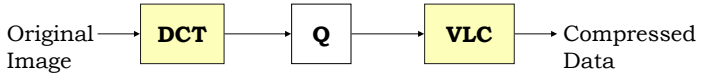


Conclusion

- goal of motion estimation is to reducing the temporal redundancy
- several motion estimation algorithms and matching criterions
- one-dimensional systolic array
- a scalable module-based motion estimation architecture is presented

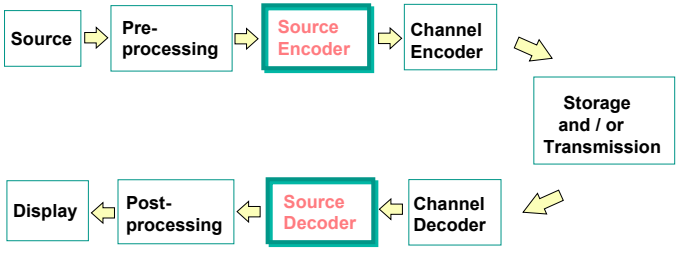
ISO/IEC JPEG

- Still image compression
- DCT, RLC, VLC, Predictive DC
- Loss compression
- Compression ratio: 8 ~ 10



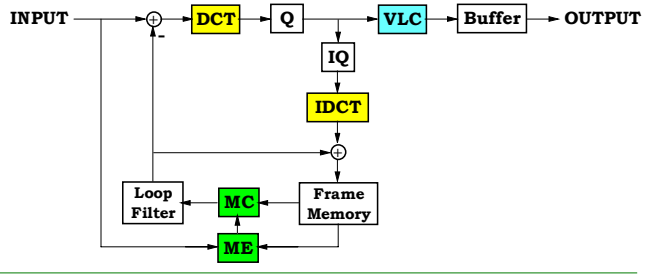
Video Compression

Video Coding and Decoding Processing



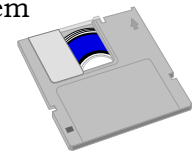
ITU-T H.261

- Video conference
- Bitrate: px64k, p=1~31



ISO/IEC MPEG-1

- Media storage
- Optimal for frame size 352x240x30 or 352x288x25
- Bitrate: up to 1.5 Mbit/s
- International standard in 1992
- Single chip for the whole system



ISO/IEC MPEG-2

- Applications from storage to HDTV
- Bitrate: standard definition TV: 4-9 Mbit/s
  HDTV: 15-25 Mbit/s
- Interlaced/non-interlaced
- Scalability
- Capable of decoding MPEG-1 bitstream
- International standard in 1994
- Single chip for video and audio

ITU-T H.263 & H.263++

- Video telephony in PSTN and mobile
- Optimized H.261/MPEG at bit-rate < 22 kbps
- Efficient on network transmission
- Optional modes
- Single chip for video and system
- Chip set for audio

ISO/IEC MPEG-4

- Applications for multimedia communication
- Bitrate: 10K-25 Mbit/s
- Object-based coding
- Natural and Synthetic video
- Scalability
- Robust and error resilience
- International standard in 1998
- Single chip for video and audio

Current Standards

標準名稱	應用	成爲國際標準時間
JPEG	still image	1990年
H.261	video conferences	1990年
MPEG-1	media storage	1992年
MPEG-2	storage-based multimedia systems， television broadcasting， video-on-demand， face-to-face communication。	1994年
H.263	visual telephoning	1995年
MPEG-4	very low bit-rate video coding	1998年*