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

<table>
<thead>
<tr>
<th>Format</th>
<th>Data Amount</th>
<th>Bit-Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>352x240 CIF</td>
<td>30.4 Mbits/s</td>
<td>1.5 Mbits/s</td>
</tr>
<tr>
<td>720x480 CCIR 601</td>
<td>165.9 Mbits/s</td>
<td>15-25 Mbits/s</td>
</tr>
<tr>
<td>1920x1080 HDTV</td>
<td>995.3 Mbits/s</td>
<td>80 Mbits/s</td>
</tr>
</tbody>
</table>
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 data at 20.736 Mbytes/sec only 31 seconds of video be stored on 650MByte CD-ROM.

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 data at 20.736 Mbytes/sec only 31 seconds of video be stored on 650MByte CD-ROM.

Storage Requirements for Multimedia applications

<table>
<thead>
<tr>
<th>OBJECT TYPE</th>
<th>TEXT</th>
<th>IMAGE</th>
<th>AUDIO</th>
<th>ANIMATION</th>
<th>VIDEO</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>ASCII</td>
<td>TIFF</td>
<td>ASCII</td>
<td>AVI</td>
<td>AVI</td>
</tr>
<tr>
<td>EBCDIC</td>
<td>EBCDIC</td>
<td>GIF</td>
<td>EBCDIC</td>
<td>MOV</td>
<td>MOV</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SIZE AND BANDWIDTH</th>
<th>TEXT (bit per page)</th>
<th>IMAGE (Mbit per Mb)</th>
<th>AUDIO (Mbit per Mb)</th>
<th>ANIMATION (Mbit per Mb)</th>
<th>VIDEO (Mbit per Mb)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple (gray scale)</td>
<td>768</td>
<td>4.4</td>
<td>41.9</td>
<td>16.1</td>
<td>10.3</td>
</tr>
<tr>
<td>Reduced (color) 5 MB per image</td>
<td>2384</td>
<td>0.6</td>
<td>3.9</td>
<td>1.6</td>
<td>1.3</td>
</tr>
</tbody>
</table>

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 group 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
  - Statistical (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**:
  - Image → Prediction or Reversible Transform → Entropy Coder → Compressed Bitstream

- **Lossy**:
  - Image → For Decorrelation & Energy Compaction → Lossless Compression
  - Image → Transform → Quantizer → Entropy Coder → Compressed Bitstream

Tradeoffs in Lossless Compression

- **Coding Efficiency** - compression ratio
- **Coding Complexity** - memory requirements, power requirements, operations per second
- **Coding Delay**

Tradeoffs in Lossy Compression

- **Signal Quality** - bit error probability, signal / noise, mean opinion score
- **Coding Efficiency** - compression ratio
- **Coding Complexity** - memory requirements, power requirements, operations per second
- **Coding Delay**
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 (bps: 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 $x_i$:

\[
\begin{align*}
\text{Predictor} & \quad \begin{array}{c}
- \\
+ \\
\end{array} \\
\end{align*}
\]

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:

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

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

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

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

![Quantizer Diagram]

- **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 quantizer.

### 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

### Temporal Redundancy

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

### Coding of Moving Pictures

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

- H.261
- MPEG
- H.263
Motion Compensation
- Motion Estimation: Block matching
- Motion Compensation: Code prediction error

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

<table>
<thead>
<tr>
<th>Code</th>
<th>Probability</th>
<th>Codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>0.45</td>
<td>1</td>
</tr>
<tr>
<td>S2</td>
<td>0.3</td>
<td>00</td>
</tr>
<tr>
<td>S3</td>
<td>0.15</td>
<td>010</td>
</tr>
<tr>
<td>S4</td>
<td>0.1</td>
<td>011</td>
</tr>
</tbody>
</table>

Arithmetic coding

Motion Compensation/Estimation

Video Coder

- Forward DCT
- Quantization
- Entropy Coding
- Inverse Quantization
- Inverse DCT
- Motion Estimation/Compensation

Marco Block

Video Sequence

Original

Reconstructed

Difference
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 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 symbol (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, $x_i$ ($i = 1 \sim N$) 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
- 2. Merge Symbols

Huffman Encoding (con’t)

- 1. information needed: prob. of symbols
- 2. Merge Symbols

Leaf node

- Binary tree ->
- $l_{avg} = 2.6 = \sum l_p$
Properties of Huffman Codes

- **Entropy of source S**:
  \[ H(S) = \eta = \sum p_i \log \frac{1}{\eta} \]
  - Average length boundary: \( \frac{1}{\log_2 \eta} < \frac{1}{\log_2 \eta} + 1 \)
  - 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 its variable length, if some bits miss, then all the data are lost (Inserting marker will improve this shortcoming)

Huffman Decoding

**Bit-Serial Decoding**
- Input: bit by bit with a fixed rate
- Output: If a codeword is found, then output the corresponding symbol and discard

**Lookup-Table-Based Decoding**
- Buffer L (max codeword length)
- Every entry should be contained, so table have \( 2^L \) entry lookup table is needed

Huffman Decoding (con’t)

- Variable input rate, constant output rate

Huffman codes with constrained Length

**Method 1**:
- Partition S into two sets S1 and S2 as
  \[ S_i = \{ s | p_i \geq \frac{1}{2} \} \]
  \[ S_2 = \{ s | p_i \leq \frac{1}{2} \} \]
  \[ q = \sum_{s \in S_2} p_i \]

Assume \( L = 7 \)
\[ \frac{1}{128} = 0.0078125 \]
Huffman codes with constrained Length (cont)

- Constrained-length Huffman codewords

<table>
<thead>
<tr>
<th>Symbol</th>
<th>P(i)</th>
<th>Codeword</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0.3</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0.1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0.1</td>
<td>100</td>
</tr>
<tr>
<td>4</td>
<td>0.2</td>
<td>1000</td>
</tr>
<tr>
<td>5</td>
<td>0.1</td>
<td>10000</td>
</tr>
<tr>
<td>6</td>
<td>0.1</td>
<td>100000</td>
</tr>
<tr>
<td>7</td>
<td>0.1</td>
<td>1000000</td>
</tr>
<tr>
<td>8</td>
<td>0.1</td>
<td>10000000</td>
</tr>
<tr>
<td>9</td>
<td>0.1</td>
<td>100000000</td>
</tr>
</tbody>
</table>

- Shortcoming: No guarantee for constant symbol rate

Method 2: Ad-Hoc Design

- For a max codeword length \( L \), define a threshold \( T = 2^{-L} \)
- If \( p_i \leq T \), set \( p_i = T \)
- Then construct Huffman table

Arithmetic Coding

- Assume we want to code "flure?"
  - Huffman: 18 bits
  - Now: 16 bits
    - 0.0713343839
    - \( 2^{-9} + 2^{-7} + 2^{-10} + 2^{-10} + 2^{-16} \)

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
  - For bitonal image, use (run, value) to represent image data, can be compressed better
    
    \[
    \begin{align*}
    & (1, 1) \quad (6, 0) \quad (1, 1) \quad (1, 0) \quad (1, 1) \quad (2, 0) \\
    & \text{[white: 1, black: 0]} \\
    \end{align*}
    \]

- JBIG Compression Standard
  - Outperform Facsimile
  - Can also handle grayscale images by bit planes

Standards for Lossless Compression (cont)

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

  \[
  \begin{align*}
  y &= 0 \\
  y &= a \\
  y &= b \\
  y &= c \\
  y &= a + b - c \\
  y &= a + c - b \\
  y &= b + c - a \\
  y &= a + b + c \\
  \end{align*}
  \]

  \( a \cdot b \cdot c \) have already known to decoder

- 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 110011100 (note: 35 = 100011, it's one's complement = 011100)

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

  \[
  \begin{array}{c|c}
  \text{Category} & \text{Prediction Residual} \\
  \hline
  0 & 0 \\
  1 & 0.1 \\
  2 & 0.3 \cdot 0.2, 0.3 \cdot 0.1 \\
  3 & 0.3 \cdot 0.4, 0.1 \cdot 0.3 \\
  4 & 0.4 \cdot 0.5, 0.1 \cdot 0.4 \\
  5 & 0.5 \cdot 0.6, 0.1 \cdot 0.5 \\
  6 & 0.6 \cdot 0.7, 0.1 \cdot 0.6 \\
  7 & 0.7 \cdot 0.8, 0.1 \cdot 0.7 \\
  8 & 0.8 \cdot 0.9, 0.1 \cdot 0.8 \\
  9 & 0.9 \cdot 1.0, 0.1 \cdot 0.9 \\
  10 & 1.0 \cdot 0.1, 0.1 \cdot 1.0 \\
  11 & 0.1 \cdot 1.0, 0.1 \cdot 0.1 \\
  12 & -0.1 \cdot 1.0, 0.1 \cdot -0.1 \\
  13 & -0.2 \cdot 1.0, 0.1 \cdot -0.2 \\
  14 & -0.3 \cdot 1.0, 0.1 \cdot -0.3 \\
  15 & -0.4 \cdot 1.0, 0.1 \cdot -0.4 \\
  16 & -0.5 \cdot 1.0, 0.1 \cdot -0.5 \\
  \end{array}
  \]

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

\[ x_{ij} + e_{ij} \rightarrow \text{Quantizer} \]

\[ P_{ij} \rightarrow \text{Predictor} \]

Sample-Based Coding Example

\[ 177 + 25 \rightarrow 3 \rightarrow \text{Quantizer} \]

\[ 152 - 24 \rightarrow \text{Predictor} \]

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 Trᵗ
- If the transformation is symmetric, Y = T X Tᵗ

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
  \[ X₀₀ \beta₀₀ + X₀₁ \beta₀₂ + \ldots + X₇₇ \beta₇₇ \]

Block basis of the DCT

1-D DCT

- Pixel basis: \( e₀, e₁, e₂, \ldots, e₇ \)
- DCT basis: \( \betaₖ \) by orthogonal projection
- \( Xᵢ = \sumᵢ \langle xᵢ, \betaₖ \rangle \betaₖ \) \( k = 0..7 \)
- \( xᵢ \) means the i-th row of the block
- \( \langle xᵢ, \betaₖ \rangle = \sumⱼ xᵢⱼ \betaⱼᵏ \) = \( 0.5c(k) \sumⱼ xᵢⱼ \cos((2j + 1)k\pi/16) \)
  = \( zᵢᵏ \)

16
2-D DCT
- Cascaded by two 1-D DCT
- Row and column operation, respectively
  - \( Y = TXT^T \)
- Note: \( c(k) = \frac{1}{\sqrt{2}} \) 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
- Source → Pre-proc → DCT → Quantization
- Lossy compressed data
  - Entropy Coding

DCT-based Decoding
- Entropy Decoding → Lossy compressed data
- Inverse Quantization → IDCT → Post-proc
- Lossy output data
**Quantization/I-Quantization**
- Eyes are insensitive 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
  - \( y[i] = t[j] b + t[i] a = a ( t[j] + c t[i]) \)
  - \( y[i] = t[j] b - t[i] a = a ( t[j] - c t[i]) \)
- 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-accumulate, 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 \( V_t(p,q) = (\text{Vec}_i, \text{Vec}_j) \)
- the location in the search range \( \Omega \) 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.

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.
Motion-compensated prediction

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

\[ R = \frac{1}{2} \log_2 \frac{\sigma^2}{D} \]

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.
**R(D) Summary**

![Rate-distortion functions for various video coding schemes](image)

**Motion-compensated prediction**

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

**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
\[ V_{t}(p,q) = (V_{c1}, V_{c2}) \]
the location in the search range \(\Omega\) 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
I
P
B
B
B
P
B
B
B
B

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
Fast Search Algorithms

One-Dimensional Full Search

For \( i = -P \) to \( +P \)  
\[ \text{Vec}_i = i \min D(\text{Vec}_i,j) \]

For \( j = -P \) to \( +P \)  
\[ \text{Vec}_j = j \min D(\text{Vec}_i,j) \]

Motion Vector = \((\text{Vec}_i, \text{Vec}_j)\)

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


Performance Comparison

<table>
<thead>
<tr>
<th>Method</th>
<th>Miss America</th>
<th>Train Calendar</th>
<th>Table Tennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>2DFS</td>
<td>38.64</td>
<td>19.57</td>
<td>25.48</td>
</tr>
<tr>
<td>1DFS</td>
<td>38.39</td>
<td>19.45</td>
<td>24.55</td>
</tr>
<tr>
<td>3Step</td>
<td>38.20</td>
<td>19.40</td>
<td>24.06</td>
</tr>
<tr>
<td>Entropy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2D FS</td>
<td>3.49</td>
<td>5.93</td>
<td>5.27</td>
</tr>
<tr>
<td>1DFS</td>
<td>3.52</td>
<td>5.94</td>
<td>5.39</td>
</tr>
<tr>
<td>3Step</td>
<td>3.56</td>
<td>5.94</td>
<td>5.47</td>
</tr>
</tbody>
</table>

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)

\[ x_{s}(k,l) : \text{luminance for the location } (k,l) \text{ in } X_{s}(p,q) \]
\[ x_{s,t}(k+i,l+j) : \text{luminance for the shifted location by } i \text{ pels and } j \text{ lines within the search range} \]

Cross-Correlation Function (CCF)

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

Mean Squared Error (MSE)

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

Mean Absolute Error (MAE)

\[ MAE(i,j) = \frac{1}{N^{2}} \sum_{k=1}^{N} \sum_{l=1}^{N} |x_{s}(k,l) - x_{s,t}(k+i,l+j)| \]

Pel Difference Classification (PDC)

\[ T(k,l,i,j) = \begin{cases} 1, & |x_{s}(k,l) - x_{s,t}(k+i,l+j)| \leq \text{Threshold} \\ 0, & \text{otherwise} \end{cases} \]

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

Minimized Maximum Error (MiniMax)

\[ G(i,j) = \max |x_{s}(k,l) - x_{s,t}(k+i,l+j)| \]

Most Widely Used in hardware:

- 8-bit Subtractor
- Absolute Function
- 16-bit Accumulator

Most Widely Used in software:

- Hardware Reduction

Hardware

PE of MAE

PE of MiniMax

- 8-bit Subtractor
- Absolute Function
- 8-bit Comparator
Subjective Quality

<table>
<thead>
<tr>
<th>Salesman</th>
<th>Train &amp; Calendar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min-max</td>
<td>MAE</td>
</tr>
<tr>
<td>Min-max</td>
<td>MAE</td>
</tr>
</tbody>
</table>

Other Approaches: Pixel Subsampling for MAE calculations

<table>
<thead>
<tr>
<th>1 2 1 2 1 2 1 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 4 3 4 3 4 3 4</td>
</tr>
<tr>
<td>1 2 1 2 1 2 1 2</td>
</tr>
<tr>
<td>3 4 3 4 3 4 3 4</td>
</tr>
<tr>
<td>1 2 1 2 1 2 1 2</td>
</tr>
<tr>
<td>3 4 3 4 3 4 3 4</td>
</tr>
<tr>
<td>1 2 1 2 1 2 1 2</td>
</tr>
<tr>
<td>3 4 3 4 3 4 3 4</td>
</tr>
</tbody>
</table>

Figure 4.15: Pixel decimation for block matching in an 8 x 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 $8 \times 8$ 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

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

\[ E(V'_1(p, q)) = V'_1(p, q) \]

Frame 1

Inter-Block Prediction

\[ E(V'_1(p, q)) = V'_1(p, q) \]

Frame 1

Median Vector Prediction

\[ E(V'_1(p, q)) = \arg \text{Median} V' \]

Frame 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

\[ X(p, q) = (x_1, x_2, x_3, x_4) \]

column vector

\[ X(p, q) = (x_1, x_2, x_3, x_4) \]

row vector

\[ E(V'_1(p, q)) = \arg \text{Min} d_{\text{med}} \]

\[ V'_1 = \{ V_1(p, q - 1), V_1(p - 1, q) \} \]

Side Match

\[ d_{\text{u}} = \sum \left| X_{\text{uc}}(p, q)_1 \right| - \left| X(p, q-1) \right| \]

\[ d_{\text{l}} = \sum \left| X_{\text{lc}}(p, q)_1 \right| - \left| X(p, q+1) \right| \]

\[ d_{\text{r}} = \sum \left| X_{\text{rc}}(p, q)_1 \right| - \left| X(p+1, q) \right| \]

\[ d_{\text{d}} = \sum \left| X_{\text{dc}}(p, q)_1 \right| - \left| X(p, q+1) \right| \]

\[ d_{\text{med}} = d_{\text{u}} + d_{\text{l}} + d_{\text{r}} + d_{\text{d}} \]

\[ E(V'_1(p, q)) = \arg \text{Min} d_{\text{med}} \]

\[ V'_1 = \{ V_1(p, q - 1), V_1(p - 1, q), V_1(p + 1, q - 1) \} \]
### Computation Comparison

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Search Range</th>
<th>Computation Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Search</td>
<td>-p ~ p</td>
<td>((2p+1)^2 \times m \times n)</td>
</tr>
<tr>
<td>Inter-Frame</td>
<td>-p/2 ~ -p/2</td>
<td>((p+1)^2 \times m \times n)</td>
</tr>
<tr>
<td>Inter-Block</td>
<td>p ~ p - p/2 ~ p/2</td>
<td>[ \frac{1}{4}(2p+1)^2 + \frac{3}{4}(p+1)^2 ] \times m \times n ] + 6</td>
</tr>
<tr>
<td>Median</td>
<td>- p/2 ~ p/2</td>
<td>((p+1)^2 \times m \times n) + 6</td>
</tr>
<tr>
<td>Boundary Match</td>
<td>- p/2 ~ p/2</td>
<td>((p+1)^2 \times m \times n) + m + n</td>
</tr>
<tr>
<td>Side Match</td>
<td>- p/2 ~ p/2</td>
<td>((p+1)^2 \times m \times n) + 6(m + n)</td>
</tr>
</tbody>
</table>

### Objective Performance Measurement

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

Predicted image compared to the original image

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

#### 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

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

Block size 8x8 | 60 frames | Table Tennis Sequence
Error Entropy vs Frame Number

Motion Vector Characteristics

Motion Vector Characteristics

MV Euclidean Distance (ED)

Hit Ratio (HR)

Criterion:

<table>
<thead>
<tr>
<th>Frame Rate</th>
<th>30 frames/sec</th>
<th>15 frames/sec</th>
<th>10 frames/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithms</td>
<td>ED HR</td>
<td>ED HR</td>
<td>ED HR</td>
</tr>
<tr>
<td>Inter-Frame</td>
<td>23.52 72.92%</td>
<td>60.05 67.55%</td>
<td>74.67 62.46%</td>
</tr>
<tr>
<td>Inter-Block</td>
<td>13.05 76.88%</td>
<td>54.54 71.37%</td>
<td>72.82 67.55%</td>
</tr>
<tr>
<td>Median</td>
<td>20.63 73.33%</td>
<td>57.15 66.98%</td>
<td>82.41 60.64%</td>
</tr>
<tr>
<td>Boundary Match</td>
<td>10.08 79.70%</td>
<td>47.88 74.76%</td>
<td>66.91 70.73%</td>
</tr>
<tr>
<td>Side Match</td>
<td>9.07 50.93%</td>
<td>46.08 75.72%</td>
<td>55.18 71.86%</td>
</tr>
<tr>
<td>IDFS</td>
<td>26.11 61.97%</td>
<td>95.13 41.74%</td>
<td>222.77 31.55%</td>
</tr>
<tr>
<td>3-Step</td>
<td>23.14 50.81%</td>
<td>109.92 30.39%</td>
<td>144.47 23.80%</td>
</tr>
</tbody>
</table>

Motion Vector Distribution

Subjective Quality

Original

1DFS

2DFS

3Step

Table Tennis Sequence

Inter-Frame

Inter-Block

Boundary Match

Side Match

Table Tennis Sequence

Subjective Quality
Hierarchical Motion Estimation

1. Level 2
   - Picture size = 180 x 120, and macroblock size = 4 x 4.
   - Number of macroblocks = \( \frac{Picture\ size}{Macroblock\ size} = \frac{1,350}{4} \). At 30 fps, we have 40,500 macroblocks.
   - Search parameter = \( \lceil \frac{2}{4+1} \rceil = 4 \).
   - Number of search locations = \( (2 \times 4 + 1^2) = 81 \).
   - Number of operations per search location = macroblock size \( \times 3 = 48 \).
   - Complexity for Level 2 = \( 40,500 \times 81 \times 48 = 157.46 \) MOPS.

2. Level 1
   - Picture size = 360 x 240, and macroblock size = 8 x 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 \( \times 3 = 192 \).
   - Complexity for Level 1 = \( 40,500 \times 9 \times 192 = 69.98 \) MOPS.

3. Level 0
   - Picture size = 720 x 480, and macroblock size = 16 x 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 \( \times 3 = 768 \).
   - Complexity for Level 0 = \( 40,500 \times 72 \times 48 = 279.9 \) 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

<table>
<thead>
<tr>
<th>Method</th>
<th>Operations per Macroblock</th>
<th>Operations for pictures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full-search</td>
<td>$(9p+1)Y/3$</td>
<td>29.89 GOPS</td>
</tr>
<tr>
<td>Logarithmic</td>
<td>$(8\log p + 1)/3$</td>
<td>1.03 GOPS</td>
</tr>
<tr>
<td>PHODS</td>
<td>$(4\log p + 1)/3$</td>
<td>528.76 GOPS</td>
</tr>
<tr>
<td>Hierarchical</td>
<td>$(p+1)/3$</td>
<td>507.38 GOPS</td>
</tr>
</tbody>
</table>

Table 4.1: Computational complexity and GOPS requirements for various motion-estimation algorithms using the MAE criterion and a $[P, P]$ search range.

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

<table>
<thead>
<tr>
<th>clocks</th>
<th>reference data</th>
<th>PE0</th>
<th>PE1</th>
<th>PE2</th>
<th>PE14</th>
<th>PE15</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>b(0,0)</td>
<td>a(0,0) - b(0,0)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>b(0,1)</td>
<td>a(0,1) - b(0,1)</td>
<td>a(0,0) - b(0,1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>b(0,2)</td>
<td>a(0,2) - b(0,2)</td>
<td>a(0,1) - b(0,2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>b(1,0)</td>
<td>a(1,0) - b(1,0)</td>
<td>a(1,1) - b(1,0)</td>
<td>a(1,0) - b(1,1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>b(1,1)</td>
<td>a(1,1) - b(1,1)</td>
<td>a(1,2) - b(1,1)</td>
<td>a(1,1) - b(1,2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>b(1,2)</td>
<td>a(1,2) - b(1,2)</td>
<td>a(1,3) - b(1,2)</td>
<td>a(1,2) - b(1,3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>b(1,3)</td>
<td>a(1,3) - b(1,3)</td>
<td>a(1,4) - b(1,3)</td>
<td>a(1,3) - b(1,4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>b(1,4)</td>
<td>a(1,4) - b(1,4)</td>
<td>a(1,5) - b(1,4)</td>
<td>a(1,4) - b(1,5)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>b(1,5)</td>
<td>a(1,5) - b(1,5)</td>
<td>a(1,6) - b(1,5)</td>
<td>a(1,5) - b(1,6)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>b(1,6)</td>
<td>a(1,6) - b(1,6)</td>
<td>a(1,7) - b(1,6)</td>
<td>a(1,6) - b(1,7)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>b(1,7)</td>
<td>a(1,7) - b(1,7)</td>
<td>a(1,8) - b(1,7)</td>
<td>a(1,7) - b(1,8)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>b(1,8)</td>
<td>a(1,8) - b(1,8)</td>
<td>a(1,9) - b(1,8)</td>
<td>a(1,8) - b(1,9)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>b(1,9)</td>
<td>a(1,9) - b(1,9)</td>
<td>a(1,10) - b(1,9)</td>
<td>a(1,9) - b(1,10)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>b(1,10)</td>
<td>a(1,10) - b(1,10)</td>
<td>a(1,11) - b(1,10)</td>
<td>a(1,10) - b(1,11)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>b(1,11)</td>
<td>a(1,11) - b(1,11)</td>
<td>a(1,12) - b(1,11)</td>
<td>a(1,11) - b(1,12)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>b(1,12)</td>
<td>a(1,12) - b(1,12)</td>
<td>a(1,13) - b(1,12)</td>
<td>a(1,12) - b(1,13)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>b(1,13)</td>
<td>a(1,13) - b(1,13)</td>
<td>a(1,14) - b(1,13)</td>
<td>a(1,13) - b(1,14)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>b(1,14)</td>
<td>a(1,14) - b(1,14)</td>
<td>a(1,15) - b(1,14)</td>
<td>a(1,14) - b(1,15)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>b(1,15)</td>
<td>a(1,15) - b(1,15)</td>
<td>a(1,16) - b(1,15)</td>
<td>a(1,15) - b(1,16)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>b(2,0)</td>
<td>a(2,0) - b(2,0)</td>
<td>a(2,1) - b(2,0)</td>
<td>a(2,0) - b(2,1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>b(2,1)</td>
<td>a(2,1) - b(2,1)</td>
<td>a(2,2) - b(2,1)</td>
<td>a(2,1) - b(2,2)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>b(2,2)</td>
<td>a(2,2) - b(2,2)</td>
<td>a(2,3) - b(2,2)</td>
<td>a(2,2) - b(2,3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>b(2,3)</td>
<td>a(2,3) - b(2,3)</td>
<td>a(2,4) - b(2,3)</td>
<td>a(2,3) - b(2,4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>b(2,4)</td>
<td>a(2,4) - b(2,4)</td>
<td>a(2,5) - b(2,4)</td>
<td>a(2,4) - b(2,5)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>b(2,5)</td>
<td>a(2,5) - b(2,5)</td>
<td>a(2,6) - b(2,5)</td>
<td>a(2,5) - b(2,6)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>b(2,6)</td>
<td>a(2,6) - b(2,6)</td>
<td>a(2,7) - b(2,6)</td>
<td>a(2,6) - b(2,7)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>b(2,7)</td>
<td>a(2,7) - b(2,7)</td>
<td>a(2,8) - b(2,7)</td>
<td>a(2,7) - b(2,8)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>b(2,8)</td>
<td>a(2,8) - b(2,8)</td>
<td>a(2,9) - b(2,8)</td>
<td>a(2,8) - b(2,9)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>b(2,9)</td>
<td>a(2,9) - b(2,9)</td>
<td>a(2,10) - b(2,9)</td>
<td>a(2,9) - b(2,10)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>b(2,10)</td>
<td>a(2,10) - b(2,10)</td>
<td>a(2,11) - b(2,10)</td>
<td>a(2,10) - b(2,11)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>b(2,11)</td>
<td>a(2,11) - b(2,11)</td>
<td>a(2,12) - b(2,11)</td>
<td>a(2,11) - b(2,12)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>b(2,12)</td>
<td>a(2,12) - b(2,12)</td>
<td>a(2,13) - b(2,12)</td>
<td>a(2,12) - b(2,13)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>b(2,13)</td>
<td>a(2,13) - b(2,13)</td>
<td>a(2,14) - b(2,13)</td>
<td>a(2,13) - b(2,14)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>b(2,14)</td>
<td>a(2,14) - b(2,14)</td>
<td>a(2,15) - b(2,14)</td>
<td>a(2,14) - b(2,15)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>b(2,15)</td>
<td>a(2,15) - b(2,15)</td>
<td>a(2,16) - b(2,15)</td>
<td>a(2,15) - b(2,16)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>b(2,16)</td>
<td>a(2,16) - b(2,16)</td>
<td>a(2,17) - b(2,16)</td>
<td>a(2,16) - b(2,17)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>b(2,17)</td>
<td>a(2,17) - b(2,17)</td>
<td>a(2,18) - b(2,17)</td>
<td>a(2,17) - b(2,18)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>b(2,18)</td>
<td>a(2,18) - b(2,18)</td>
<td>a(2,19) - b(2,18)</td>
<td>a(2,18) - b(2,19)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>b(2,19)</td>
<td>a(2,19) - b(2,19)</td>
<td>a(2,20) - b(2,19)</td>
<td>a(2,19) - b(2,20)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Reference Data Access

Reference Data Access of Multiple Modules

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

- Source → Pre-processing → Source Encoder → Channel Encoder → Storage and / or Transmission → Display
- Source Decoder → Channel Decoder → Post-processing → Display

ITU-T H.261

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

DCT → Q → VLC → Buffer → OUTPUT

DCT

DCT

Frame Memory

ME

MC

Loop Filter

IQ

IDCT

INPUT
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

<table>
<thead>
<tr>
<th>Standard</th>
<th>Application</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>JPEG</td>
<td>still image</td>
<td>1990</td>
</tr>
<tr>
<td>H.261</td>
<td>video conferences</td>
<td>1990</td>
</tr>
<tr>
<td>MPEG-1</td>
<td>media storage</td>
<td>1992</td>
</tr>
<tr>
<td>MPEG-2</td>
<td>storage-based multimedia systems, television broadcasting, video-on-demand, face-to-face communication</td>
<td>1994</td>
</tr>
<tr>
<td>H.263</td>
<td>visual telephoning</td>
<td>1995</td>
</tr>
<tr>
<td>MPEG-4</td>
<td>very low bit-rate video coding</td>
<td>1998</td>
</tr>
</tbody>
</table>