MPEG-4

- Coding of audio-visual objects
- Goal – provide core technology that allows efficient content-based storage, transmission and manipulation of video, graphics, audio and other data in a multimedia environment
- Applications – Digital TV, mobile multimedia, TV production, games, streaming video
- Requirements of MPEG-4
  - Allowance of interactivity
  - High compression
  - Universal accessibility
  - Portability of audio and visual content

Introduction to MPEG-4 (cont.)
Functionality of MPEG-4

- Content-based efficient compression
- Content-based interactivity
  - Content-based manipulation and bit-stream editing in the compressed domain
  - Content-based scalability at the object level
  - Synthetic and natural hybrid coding (SNHC)
  - Improved temporal random access
- Universal access
  - Robustness in error-prone environments
  - Content-based (spatial and temporal resolution) scalability

MPEG-4 Video Hierarchy

- Video object plane (VOP)
  - Each frame is segmented into a set of arbitrarily shaped regions or VOPs
  - The input to a video encoder can be a VOP, the shape and the location of the VOP
- Group of video object planes (GOV)
  - Group together video object planes, similar to groups of pictures (GOPs) in MPEG-1/2
  - Provide random access points into the bit-stream
- Video object layer (VOL)
  - Permit scalable coding (spatial, temporal) of a sequence of VOPs or GOVs
  - Multiple VOLs correspond to multiple scaling of a sequence
MPEG-4 Video Hierarchy (cont.)

- Video object (VO)
  - Types of visual objects – video objects, texture objects, mesh objects, and face objects
- Video session (VS)
  - Include all video objects (natural and synthetic) in the scene
Video Object Coding (cont.)

- A VOP is defined by its texture (luminance and chrominance value) and shape
- Three modes for encoding an VOP
  - Intra VOP (I-VOP)
  - Predicted VOP (P-VOP)
  - Bidirectionally predicted VOP (B-VOP)
Video Object Coding (cont.)

- Encoding of a VOP consists of
  - **Motion vector coding** – motion compensated prediction on the texture information (luminance and chrominance value) to reduce temporal redundancy
  - **Texture coding** – DCT-based coding of motion compensated prediction error to reduce spatial redundancies
  - **Shape coding** – for arbitrary shaped VOPs
    - encode the binary shape and the transparency information of the object
    - Motion estimation and compensation are also performed on the shape of the object to reduce temporal redundancy
Motion Estimation and Compensation

- New features of MPEG-4 motion compensation
  - Adaptive selection of 16x16 or four 8x8 blocks
  - Unrestricted motion vectors – MVs are allowed to point outside the coded area of a reference VOP
  - Overlapped motion compensation
    - used for 8x8 block matching
    - The prediction value is given by the following equation:
      \[ p'(i, j) = \frac{H_0(i, j) \times q(i, j) + H_1(i, j) \times r(i, j) + H_2(i, j) \times s(i, j)}{8} \]
      where
      \[ q(i, j) = p(i+MV_x^0, j+MV_y^0) \]
      \[ r(i, j) = p(i+MV_x^1, j+MV_y^1) \]
      \[ s(i, j) = p(i+MV_x^2, j+MV_y^2) \]

Motion Estimation and Compensation (cont.)

- Overlapped motion compensation (cont.)
  - (MV_x^0, MV_y^0) is the MV of the current 8x8 block \(p(i, j)\)
  - (MV_x^1, MV_y^1) is the MV of the block either above (for \(j = 0, 1, 2, 3\)) or below (for \(j = 4, 5, 6, 7\)) of the current block
  - (MV_x^2, MV_y^2) is the MV of the block either to the left (for \(i = 0, 1, 2, 3\)) or right (for \(i = 4, 5, 6, 7\)) of the current block

\[
\begin{align*}
H_0 &= \begin{bmatrix}
4 & 5 & 5 & 5 & 5 & 5 & 5 & 4 \\
5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\
5 & 6 & 6 & 6 & 6 & 6 & 6 & 5 \\
5 & 5 & 6 & 6 & 6 & 6 & 6 & 5 \\
5 & 5 & 6 & 6 & 6 & 6 & 6 & 5 \\
5 & 5 & 6 & 6 & 6 & 6 & 6 & 5 \\
5 & 5 & 5 & 5 & 5 & 5 & 5 & 4 \\
4 & 5 & 5 & 5 & 5 & 5 & 5 & 4
\end{bmatrix} & H_1 &= \begin{bmatrix}
2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\
1 & 1 & 2 & 2 & 2 & 2 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 2 & 2 & 2 & 2 & 1 & 1 \\
2 & 2 & 2 & 2 & 2 & 2 & 2 & 2
\end{bmatrix} & H_2 &= \begin{bmatrix}
2 & 1 & 1 & 1 & 1 & 1 & 1 & 2 \\
2 & 2 & 1 & 1 & 1 & 1 & 2 & 2 \\
2 & 2 & 1 & 1 & 1 & 1 & 2 & 2 \\
2 & 2 & 1 & 1 & 1 & 1 & 2 & 2 \\
2 & 2 & 1 & 1 & 1 & 1 & 2 & 2 \\
2 & 2 & 1 & 1 & 1 & 1 & 2 & 2 \\
2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\
2 & 1 & 1 & 1 & 1 & 1 & 1 & 2
\end{bmatrix} 
\end{align*}
\]
Motion Estimation and Compensation (cont.)

- Unrestricted motion vectors

Reference I-VOP or P-VOP

\[ \text{modified block (polygon) matching} \]

Reference VOP pixels for block matching

Padded reference pixels for block matching

P-VOP or B-VOP

\[ \text{no matching} \]

Texture Coding

- For coding intrablocks and motion compensation prediction error blocks

VOP texture → DCT → Quantization → Coefficient Scan → Coefficient Prediction

Coefficient Scan

Variable Length Coding → Bit stream
Texture Coding (cont.)

- Adaptive DC prediction
  - The predictor compares the horizontal and vertical DC value gradient and predict the DC value from the block above or to the left in the direction of lesser gradient:
    \[ Q_{DC} = \frac{dc}{8} \]
    \[ \text{if}(|Q_{DC_A} - Q_{DC_B}| < |Q_{DC_B} - Q_{DC_C}|) \]
    \[ Q_{DC_X} = Q_{DC_C} \]
    \[ \text{else} \]
    \[ Q_{DC_X} = Q_{DC_A} \]

Texture Coding (cont.)

- Adaptive AC coefficients prediction
  - The AC coefficients of either the first row or the first column are predicted from those of the block immediately above or to the left
Texture Coding (cont.)

- **DCT coefficients scanning order**
  - Zigzag scan – used when no DC prediction
  - Alternate-vertical scan – if DC was predicted from left block
  - Alternate-horizontal scan – if DC was predicted from the block above

<table>
<thead>
<tr>
<th>Zigzag scan</th>
<th>Alternate-vertical scan</th>
<th>Alternate-horizontal scan</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63</td>
<td>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63</td>
</tr>
</tbody>
</table>

Texture Coding (cont.)

- **Texture coding of arbitrarily shaped VOP**
  - Inside block – macroblocks that are completely located inside of the VOP, coded with 8×8 DCT with motion compensation
  - Outside block – macroblocks outside of the boundary, need no texture coding
  - Boundary blocks – located along the boundary of the VOP
Texture Coding (cont.)

- Texture coding of arbitrarily shaped VOP (cont.)
  - Two kinds of 8×8 blocks the boundary macroblocks:
    1. Transparent blocks – the blocks that do not belong to the arbitrary shape but lie inside the rectangular bounding box of the VOP, not coded
    2. Boundary blocks – the blocks lie along the boundary of VOP, coded by
      - Low-pass extrapolation (LPE) padding
      - Shape-adaptive DCT

Texture Coding (cont.)

- Low-pass extrapolation (LPE) padding – for intra blocks
  1. Calculate the arithmetic mean value $\bar{m}$ of all block pixels situated within the object region
  2. Assign $\bar{m}$ to each block pixel located outside of the object region
  3. Apply the following filtering operation to each block pixel outside of the object region
     \[ p(i, j) = \frac{1}{4} [ p(i, j-1) + p(i-1, j) + p(i, j+1) + p(i+1, j)] \]
     if one or more of the four pixels used for filtering are outside of the block, the corresponding pixels are not considered for the average operation and the factor 1/4 is modified accordingly
Texture Coding (cont.)

- Shape-adaptive DCT (SA-DCT)
  - Apply 1-D DCT vertically according to the number of active pixels in the column of the block
  - The coefficients of the vertical DCTs with the same frequency index are lined up in a row
  - Apply 1-D DCT horizontally to each row
  - All SA-DCT coefficients are quantized and VLC coded

Interlaced Coding

- Frame DCT coding - each block shall be composed of lines from the two fields alternately
- Field DCT coding - each block shall be composed of lines from only one of the two fields
Texture Coding (cont.)

- Static texture coding – zero-tree based wavelet coding
  1. Decompose the texture using discrete wavelet transform (DWT)
  2. Quantize the wavelet coefficients
  3. Coding the lowest frequency subband using a predictive scheme
  4. Zero-tree coding the higher order subband wavelet coefficients

Shape Coding

- Binary alpha plane – define the shape of a VOP, indicate whether or not a pixel belongs to a VOP
- Gray-scale alpha plane – define the transparency of each pixel within a VOP
- VOP enclosed in a rectangular bounding box (mask) and divided into 16×16 macroblocks
Binary Shape Coding

- Binary alpha block (BAB) – each 16×16 block within the rectangular box and can be classified as three classes
  - Transparent block – all pixels are outside the object
  - Opaque block – all pixels are inside the object
  - Alpha (shape) block – some are transparent and others opaque
    - Encoded by modified content-based arithmetic encoding (CAE)

![Diagram of Binary Shape Coding]

Binary Shape Coding (cont.)

- Intra shape coding
  1. Compute a shape context number based on a template
     \[ C = \sum_{k=0}^{9} c_k 2^k \]
     
     \[ c_k = 0 \text{ for transparent pixels and } 1 \text{ for opaque pixels} \]
  2. Use this context number to index a probability table
  3. Use the indexed probability to drive a binary arithmetic encoder

![Diagram of Intra Shape Coding]
Binary Shape Coding (cont.)

- **Inter shape coding**
  1. The shape of the current block is predicted using motion estimation and compensation in integer pixel accuracy
  2. The motion vectors is coded predictively
  3. The difference between the current and the predicted shape block is arithmetically coded

\[
C = \sum_{k=0}^{8} c_k 2^k
\]

<table>
<thead>
<tr>
<th>c_0</th>
<th>c_1</th>
<th>c_2</th>
<th>c_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>c_4</td>
<td>c_5</td>
<td>c_6</td>
</tr>
</tbody>
</table>

- **Coding modes**
  1. Transparent
  2. Opaque
  3. Intra
  4. Inter with shape motion vectors
  5. Inter without shape motion vectors
  6. Inter with shape motion vectors and prediction error coding
  7. Inter without shape motion vectors and predictive error coding
**Binary Shape Coding (cont.)**

- Lossy shape coding
  1. Not transmitting the difference between the current and the predicted shape block (in inter shape coding)
  2. **Subsampling** the binary alpha plane by a factor of 2 or 4 prior to arithmetic encoding (in both intra and inter coding)
     - An upsampling filter is employed during the reconstruction to reduce the blocky appearance

**Gray Scale Shape Coding**

- Gray scale shape information contains two parts
  1. Support function
     - the shape information
     - encoded using the binary shape coding method
  2. Transparency information
     - the alpha values – the texture of luminance
     - encoded using padding, motion compensation, 8x8 DCT approach for texture coding
Sprite Coding

- **Sprite** – a special video object (like a static background) that is visible through an entire piece of video sequence
  - **Static sprite** – a possible still image
    - used for a video sequence in which the objects in a scene can be separated into foreground objects and background objects
    - well suited for **synthetic objects** and objects that mostly undergo **rigid motion**
    - The changes within the background contents are mainly caused by **camera parameters**
    - generated off-line
  - **Dynamic sprite**
    - the sprite is dynamically built during the predictive coding

Sprite Coding (cont.)
Scalability

- MPEG-4 provides both spatial and temporal scalability
- The bit stream consists of a separately decodable base layer and associated enhancement layers

Scalability (cont.)

- Spatial scalability
  - The base layer VOPs are coded as conventional MPEG-4 video
  - The VOPs in enhancement layer are coded as P-VOP or B-VOP
    - The current VOP in the enhancement layer can be predicted from
      - the up-sampled base layer VOP
      - the previously decoded VOP at the same layer
Scalability (cont.)

- Temporal scalability
  - Type 1: only a portion of the VOP in the base layer is enhanced
  - Type 2: the entire VOP in the base layer is enhanced

<table>
<thead>
<tr>
<th>Enhancement type 1</th>
<th>Base layer</th>
<th>enhancement layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOL0 : entire frame</td>
<td>VOL1 : car</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Enhancement type 2</th>
<th>Base layer</th>
<th>enhancement layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOL0 : entire frame</td>
<td>VOL1 : entire frame</td>
<td></td>
</tr>
</tbody>
</table>

VOL0 : car
VOL1 : entire frame

: region to be enhanced by an enhancement layer

Scalability (cont.)

- Temporal scalability - Type 1 Enhancement with P-VOPs
Scalability (cont.)

- Temporal scalability - Type 1 Enhancement with B-VOPs

Scalability (cont.)

- Temporal scalability - Type 2 Enhancement