多媒体编码及其信息安全应用

Multimedia Coding and Its Application to Information Security

第六讲

H.264关键技术及H.265简介

授课时间：2022年3月28日

内 容 提 纲 （3 节 课 内 容）

1. H.264/AVC编码标准特性

2. H.265/HEVC简介

3. 小结

内 容 提 纲 （3 节 课 内 容）

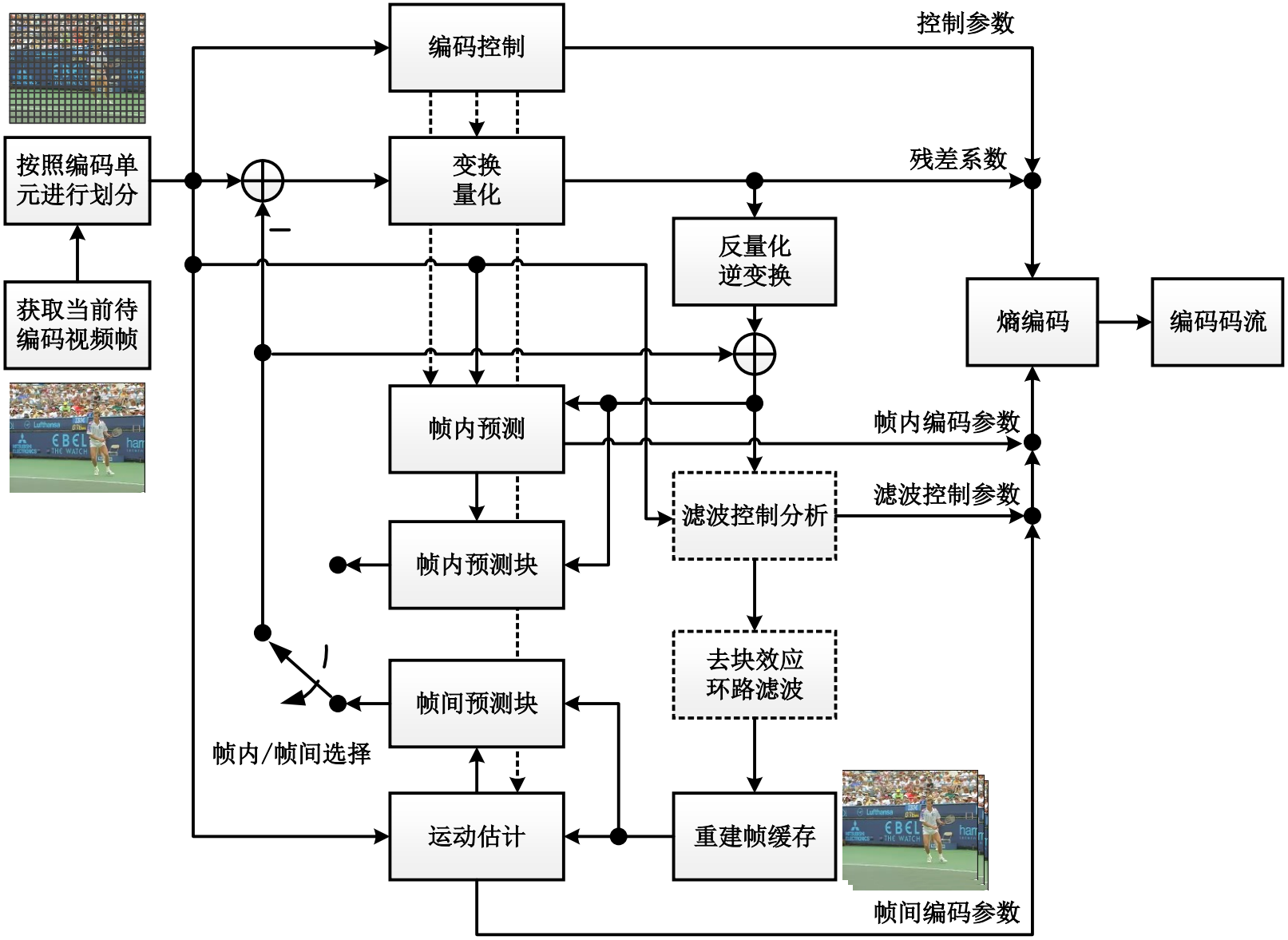
1. H.264/AVC编码标准特性

2. H.265/HEVC简介

3. 小结

H.264/AVC视频编码框架

H.264编码器框架



1.1 变换与量化

变换编码技术简介

- 变换（Transform）编码将视频的空域信号变换为频域信号，使大部分能量集中到低频区域。从而有选择地编码人眼敏感的低频信号，部分丢弃人眼不敏感的高频信号，以此提高压缩效率。

- 自1968 年快速傅立叶变换（Fast Fourier Transform，FFT）技术被利用进行图像编码以来，出现了多种正交变换编码方法，如K-L（Karhunen-Loeve）变换、离散余弦变换（Discrete Cosine Transform，DCT）等。

- K-L变换（又称为主成分变换、特征向量变换）是**均方误差**标准下的最佳变换。但其变换矩阵并不是确定的，而是由其所处理的图像决定的，所以K-L 变换矩阵需要根据图像进行统计更新，因此计算复杂度非常高，并且没有快速算法，很难在实际应用中被采用。

- 离散余弦变换编码性能接近于K-L变换，并且具有快速算法，广泛应用于图像编码。JPEG 图像压缩就是基于8×8分块（浮点）离散余弦变换技术来进行变换编码的。

1.1变换与量化

一维离散余弦变换

- **1-D DCT** 将一维N个离散实数采样值变换为N个变换域上的变换系数的过程。定义为：

$$F(u)=\sqrt{\frac{2}{N}}c(u)\sum_{n=0}^{N-1}f(n)\cos\frac{(2n+1)u\pi}{2N}\quad(u=0,1,\ldots,N-1)$$

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

其中，$f(n)$为一维离散采样值，$F(n)$为一维变换系数。

- **1-D IDCT** 将N个变换系数重构为N个离散实数采样值的过程。定义为：

$$f(n)=\sqrt{\frac{2}{N}}\sum_{u=0}^{N-1}c(u)F(u)\cos\frac{(2n+1)u\pi}{2N}\quad(n=0,1,\ldots,N-1)$$

1.1变换与量化

一维离散余弦变换

- 当N取8时

$$F(u)=\frac{1}{2}c(u)\sum_{n=0}^{7}f(n)\cos\frac{(2n+1)u\pi}{16}\quad(u=0,1,\ldots,7)$$

$$\begin{pmatrix}F(0)\\F(1)\\\vdots\\F(7)\end{pmatrix}=\begin{bmatrix}\frac{c(0)}{2}&\cdots&0\\\vdots&\ddots&\vdots\\0&\cdots&\frac{c(7)}{2}\end{bmatrix}\begin{bmatrix}\cos\frac{1\cdot0}{16}\pi&\cos\frac{3\cdot0}{16}\pi&\cdots&\cos\frac{15\cdot0}{16}\pi\\\cos\frac{1\cdot1}{16}\pi&\cos\frac{3\cdot1}{16}\pi&&\cos\frac{15\cdot1}{16}\pi\\&\vdots&\ddots&\vdots\\\cos\frac{1\cdot7}{16}\pi&\cos\frac{3\cdot7}{16}\pi&\cdots&\cos\frac{15\cdot7}{16}\pi\end{bmatrix}\begin{pmatrix}f(0)\\f(1)\\\vdots\\f(7)\end{pmatrix}$$

$$\begin{pmatrix}F(0)\\F(1)\\\vdots\\F(7)\end{pmatrix}=\frac{1}{\sqrt{8}}\begin{bmatrix}1&1&&1\\\sqrt{2}\cos\frac{1}{16}\pi&\sqrt{2}\cos\frac{3}{16}\pi&\cdots&\sqrt{2}\cos\frac{15}{16}\pi\\&\vdots&\ddots&\vdots\\\sqrt{2}\cos\frac{7}{16}\pi&\sqrt{2}\cos\frac{21}{16}\pi&\cdots&\sqrt{2}\cos\frac{105}{16}\pi\end{bmatrix}\begin{pmatrix}f(0)\\f(1)\\\vdots\\f(7)\end{pmatrix}$$

一维离散余弦变换

- 当N取8时

$$\mathbf{C}_8=\frac{1}{\sqrt{8}}\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \sqrt{2}\cos\frac{1}{16}\pi & \sqrt{2}\cos\frac{3}{16}\pi & \cdots & \sqrt{2}\cos\frac{15}{16}\pi \\ \vdots & & \ddots & \vdots \\ \sqrt{2}\cos\frac{7}{16}\pi & \sqrt{2}\cos\frac{21}{16}\pi & \cdots & \sqrt{2}\cos\frac{105}{16}\pi \end{bmatrix}$$

$$\begin{pmatrix} F(0) \\ F(1) \\ \vdots \\ F(7) \end{pmatrix}=\mathbf{C}_8\begin{pmatrix} f(0) \\ f(1) \\ \vdots \\ f(7) \end{pmatrix}$$

$$\begin{pmatrix} f(0) \\ f(1) \\ \vdots \\ f(7) \end{pmatrix}=\mathbf{C}_8^{-1}\begin{pmatrix} F(0) \\ F(1) \\ \vdots \\ F(7) \end{pmatrix}=\mathbf{C}_8^{\mathrm{T}}\begin{pmatrix} F(0) \\ F(1) \\ \vdots \\ F(7) \end{pmatrix}$$

1.1变换与量化

二维离散余弦变换

- **2-D DCT** 将二维$N \times M$个离散实数采样值变换为$N \times M$个变换域上的变换系数的过程。定义为：

$$F(u,v) = \sqrt{\frac{2}{N}}\sqrt{\frac{2}{M}}c(u)c(v)\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}f(n,m)\cos\frac{(2n+1)u\pi}{2N}\cos\frac{(2m+1)v\pi}{2M}$$

$$(u = 0,1,\dots,N-1.\ v = 0,1,\dots,M-1)$$

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

其中，$f(n,m)$为二维离散采样值，$F(n,m)$为二维变换系数。

- **2-D IDCT** 将$N \times M$个变换系数重构为$N \times M$个离散实数采样值的过程。定义为：

$$f(n,m) = \sqrt{\frac{2}{N}}\sqrt{\frac{2}{M}}\sum_{u=0}^{N-1}\sum_{v=0}^{M-1}c(u)c(v)\,F(u,v)\cos\frac{(2n+1)u\pi}{2N}\cos\frac{(2m+1)v\pi}{2M}$$

$$(n = 0,1,\dots,N-1.\ m = 0,1,\dots,M-1)$$

二维离散余弦变换

- 当N, M取8时

$$\mathbf{C}_8 = \frac{1}{\sqrt{8}}\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \sqrt{2}\cos\frac{1}{16}\pi & \sqrt{2}\cos\frac{3}{16}\pi & \cdots & \sqrt{2}\cos\frac{15}{16}\pi \\ \vdots & & \ddots & \vdots \\ \sqrt{2}\cos\frac{7}{16}\pi & \sqrt{2}\cos\frac{21}{16}\pi & \cdots & \sqrt{2}\cos\frac{105}{16}\pi \end{bmatrix}$$

$$\mathbf{Y} = \mathbf{C}_8\mathbf{X}\mathbf{C}_8^{\mathrm{T}}$$

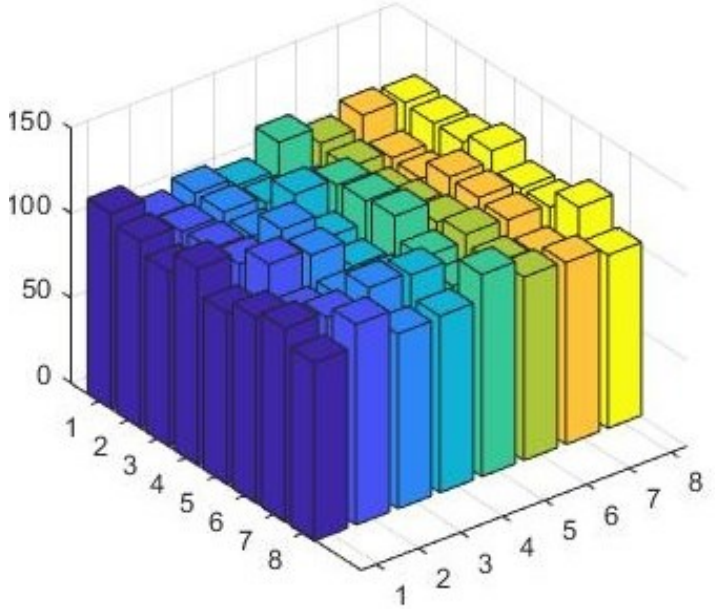
$$\mathbf{X} = \mathbf{C}_8^{\mathrm{T}}\mathbf{Y}\mathbf{C}_8$$

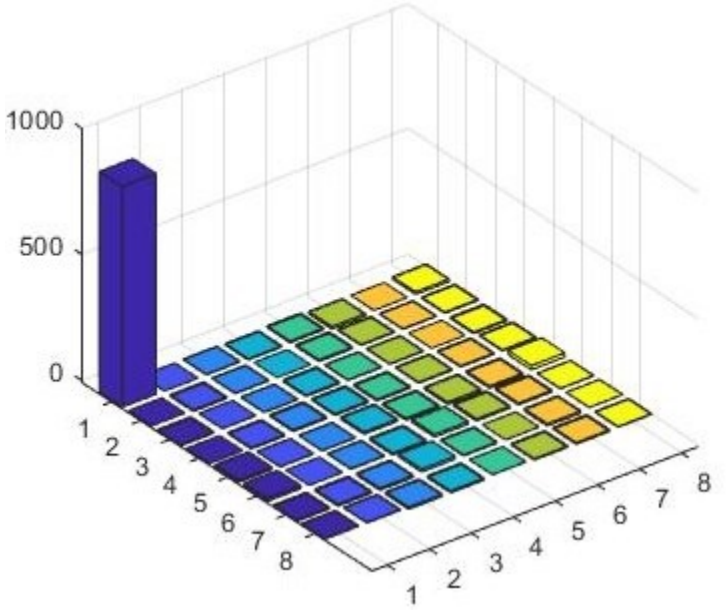
1.1变换与量化

115	103	108	103	119	109	116	114
110	106	108	104	100	106	102	113
103	107	105	120	116	101	101	113
116	109	120	105	118	105	113	119
101	120	116	100	120	107	112	108
109	102	103	102	107	111	110	108
114	111	119	117	101	105	101	119
105	118	103	104	119	108	107	102

DCT

875	-1	3	-5	7	-10	0	10
0	-6	3	0	7	4	3	4
-5	1	2	6	3	3	0	-6
0	-6	7	6	-7	4	5	5
11	-2	-2	9	-8	-9	-9	17
12	0	3	-8	11	8	7	2
-5	-8	-9	12	5	-1	-9	1
3	3	-9	-8	1	-5	-6	-3





1.1变换与量化

H.264/AVC中的整数离散余弦变换

- 离散余弦变换也可以表示成矩阵运算的形式

$$\mathbf{Y} = \mathbf{C}_N \mathbf{X} \mathbf{C}_N^{\mathrm{T}}$$

$$\mathbf{C}_N(i,j) = \sqrt{\frac{2}{N}} c(i) \cos\left[\frac{(2j+1)i}{2N}\pi\right]$$

- 在H.264中对4×4的待编码系数块进行操作，则相应的变换矩阵为

$$\mathbf{C}_4 = \begin{bmatrix} \frac{1}{2}\cos(0) & \frac{1}{2}\cos(0) & \frac{1}{2}\cos(0) & \frac{1}{2}\cos(0) \\ \sqrt{\frac{1}{2}}\cos(\frac{\pi}{8}) & \sqrt{\frac{1}{2}}\cos(\frac{3\pi}{8}) & \sqrt{\frac{1}{2}}\cos(\frac{5\pi}{8}) & \sqrt{\frac{1}{2}}\cos(\frac{7\pi}{8}) \\ \sqrt{\frac{1}{2}}\cos(\frac{2\pi}{8}) & \sqrt{\frac{1}{2}}\cos(\frac{6\pi}{8}) & \sqrt{\frac{1}{2}}\cos(\frac{10\pi}{8}) & \sqrt{\frac{1}{2}}\cos(\frac{14\pi}{8}) \\ \sqrt{\frac{1}{2}}\cos(\frac{3\pi}{8}) & \sqrt{\frac{1}{2}}\cos(\frac{9\pi}{8}) & \sqrt{\frac{1}{2}}\cos(\frac{15\pi}{8}) & \sqrt{\frac{1}{2}}\cos(\frac{21\pi}{8}) \end{bmatrix}$$

H.264/AVC整数离散余弦变换

- 令$a=\frac{1}{2}\cos(0), b=\sqrt{\frac{1}{2}}\cos(\frac{\pi}{8}), c=\sqrt{\frac{1}{2}}\cos(\frac{3\pi}{8})$，则有

$$\mathbf{C}_4=\begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix}$$

- $\mathbf{C}_4$可进一步分解为

$$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & c/b & -c/b & -1 \\ 1 & -1 & -1 & 1 \\ c/b & -1 & 1 & -c/b \end{bmatrix}=\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & d & -d & -1 \\ 1 & -1 & -1 & 1 \\ d & -1 & 1 & -d \end{bmatrix}$$

其中，$d=\frac{c}{b}=\frac{\cos\frac{3\pi}{8}}{\cos(\frac{\pi}{8})}=\tan\frac{\pi}{8}=\sqrt{2}-1=0.414$.为了将上式整数化，令$d=\frac{1}{2}$则

$$\mathbf{C}_4=\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix}$$

H.264/AVC整数离散余弦变换

- 将右边矩阵的1/2因子移到左边矩阵，可得

$$\mathbf{C}_4=\begin{bmatrix}a & 0 & 0 & 0\\ 0 & b/2 & 0 & 0\\ 0 & 0 & a & 0\\ 0 & 0 & 0 & b/2\end{bmatrix}\begin{bmatrix}1 & 1 & 1 & 1\\ 2 & 1 & -1 & -2\\ 1 & -1 & -1 & 1\\ 1 & -2 & 2 & -1\end{bmatrix}$$

为了维持变换矩阵的正交性，需满足$2a^2=b^2+c^2=(d^2+1)b^2$，可得$b=\sqrt{\frac{2a^2}{d^2+1}}$.由$a=d=\frac{1}{2}$可得$b=\sqrt{\frac{2}{5}}$.

- 整数离散余弦变换过程为

$$\mathbf{Y}=\mathbf{C}_4\mathbf{X}\mathbf{C}_4^{\mathrm{T}}=$$

$$\begin{bmatrix}a & 0 & 0 & 0\\ 0 & b/2 & 0 & 0\\ 0 & 0 & a & 0\\ 0 & 0 & 0 & b/2\end{bmatrix}\begin{bmatrix}1 & 1 & 1 & 1\\ 2 & 1 & -1 & -2\\ 1 & -1 & -1 & 1\\ 1 & -2 & 2 & -1\end{bmatrix}\mathbf{X}\begin{bmatrix}1 & 2 & 1 & 1\\ 1 & 1 & -1 & -2\\ 1 & -1 & -1 & 2\\ 1 & -2 & 1 & -1\end{bmatrix}\begin{bmatrix}a & 0 & 0 & 0\\ 0 & b/2 & 0 & 0\\ 0 & 0 & a & 0\\ 0 & 0 & 0 & b/2\end{bmatrix}=$$

$$\left(\begin{bmatrix}1 & 1 & 1 & 1\\ 2 & 1 & -1 & -2\\ 1 & -1 & -1 & 1\\ 1 & -2 & 2 & -1\end{bmatrix}\mathbf{X}\begin{bmatrix}1 & 2 & 1 & 1\\ 1 & 1 & -1 & -2\\ 1 & -1 & -1 & 2\\ 1 & -2 & 1 & -1\end{bmatrix}\right)\otimes\begin{bmatrix}a^2 & ab/2 & a^2 & ab/2\\ ab/2 & b^2/4 & ab/2 & b^2/4\\ a^2 & ab/2 & a^2 & ab/2\\ ab/2 & b^2/4 & ab/2 & b^2/4\end{bmatrix}$$

H.264/AVC整数离散余弦变换

- 根据$a=\frac{1}{2}, b=\sqrt{\frac{2}{5}}$可得

$$\mathbf{C}_4\mathbf{X}\mathbf{C}_4^{\mathrm{T}}=\left(\begin{bmatrix}1&1&1&1\\2&1&-1&-2\\1&-1&-1&1\\1&-2&2&-1\end{bmatrix}\mathbf{X}\begin{bmatrix}1&2&1&1\\1&1&-1&-2\\1&-1&-1&2\\1&-2&1&-1\end{bmatrix}\right)\otimes\begin{bmatrix}a^2&\frac{ab}{2}&a^2&\frac{ab}{2}\\\frac{ab}{2}&\frac{b^2}{4}&\frac{ab}{2}&\frac{b^2}{4}\\a^2&\frac{ab}{2}&a^2&\frac{ab}{2}\\\frac{ab}{2}&\frac{b^2}{4}&\frac{ab}{2}&\frac{b^2}{4}\end{bmatrix}=$$

$$\left(\begin{bmatrix}1&1&1&1\\2&1&-1&-2\\1&-1&-1&1\\1&-2&2&-1\end{bmatrix}\mathbf{X}\begin{bmatrix}1&2&1&1\\1&1&-1&-2\\1&-1&-1&2\\1&-2&1&-1\end{bmatrix}\right)\otimes\begin{bmatrix}\frac{1}{4}&\frac{1}{2\sqrt{10}}&\frac{1}{4}&\frac{1}{2\sqrt{10}}\\\frac{1}{2\sqrt{10}}&\frac{1}{10}&\frac{1}{2\sqrt{10}}&\frac{1}{10}\\\frac{1}{4}&\frac{1}{2\sqrt{10}}&\frac{1}{4}&\frac{1}{2\sqrt{10}}\\\frac{1}{2\sqrt{10}}&\frac{1}{10}&\frac{1}{2\sqrt{10}}&\frac{1}{10}\end{bmatrix}$$

1.1变换与量化

H.264/AVC整数离散余弦变换

- 整数离散余弦逆变换过程为

$$\mathbf{X}=\mathbf{C}_4^{\mathrm{T}}\mathbf{Y}\mathbf{C}_4=$$

$$\begin{bmatrix}1&1&1&d\\1&d&-1&-1\\1&-d&-1&1\\1&-1&1&-d\end{bmatrix}\begin{bmatrix}a&0&0&0\\0&b&0&0\\0&0&a&0\\0&0&0&b\end{bmatrix}\mathbf{Y}\begin{bmatrix}a&0&0&0\\0&b&0&0\\0&0&a&0\\0&0&0&b\end{bmatrix}\begin{bmatrix}1&1&1&1\\1&d&-d&-1\\1&-1&-1&1\\d&-1&1&-d\end{bmatrix}$$

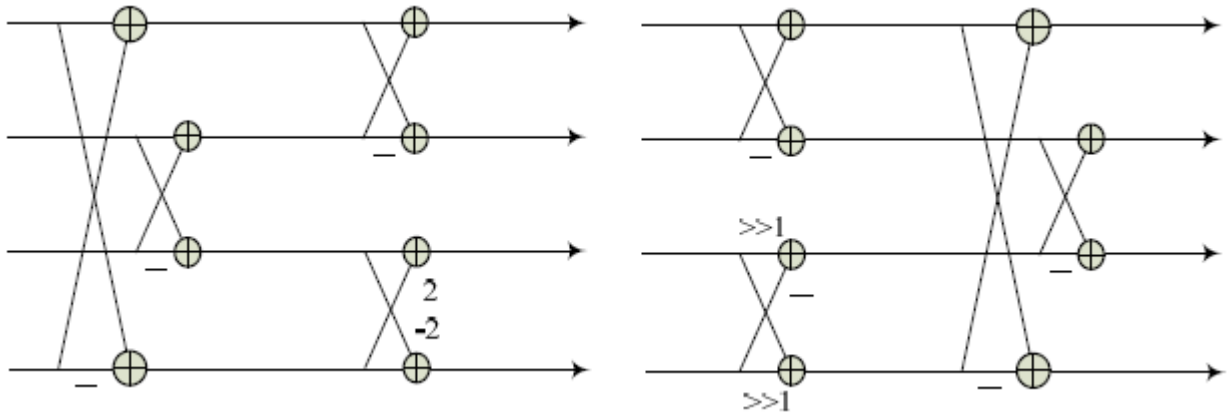
$$=\begin{bmatrix}1&1&1&1/2\\1&1/2&-1&-1\\1&-1/2&-1&1\\1&-1&1&-1/2\end{bmatrix}\begin{bmatrix}a&0&0&0\\0&b&0&0\\0&0&a&0\\0&0&0&b\end{bmatrix}\mathbf{Y}\begin{bmatrix}a&0&0&0\\0&b&0&0\\0&0&a&0\\0&0&0&b\end{bmatrix}\begin{bmatrix}1&1&1&1\\1&1/2&-1/2&-1\\1&-1&-1&1\\1/2&-1&1&-1/2\end{bmatrix}$$

$$=\begin{bmatrix}1&1&1&1/2\\1&1/2&-1&-1\\1&-1/2&-1&1\\1&-1&1&-1/2\end{bmatrix}\left(\mathbf{Y}\otimes\begin{bmatrix}a^2&ab&a^2&ab\\ab&b^2&ab&b^2\\a^2&ab&a^2&ab\\ab&b^2&ab&b^2\end{bmatrix}\right)\begin{bmatrix}1&1&1&1\\1&1/2&-1/2&-1\\1&-1&-1&1\\1/2&-1&1&-1/2\end{bmatrix}$$

1.1变换与量化

$$\mathbf{C}_4\begin{pmatrix}a\\b\\c\\d\end{pmatrix}=\begin{bmatrix}1&1&1&1\\2&1&-1&-2\\1&-1&-1&1\\1&-2&2&-1\end{bmatrix}\begin{pmatrix}a\\b\\c\\d\end{pmatrix}=\begin{pmatrix}a+b+c+d\\2a+b-c-2d\\a-b-c+d\\a-2b+2c-d\end{pmatrix}=\begin{pmatrix}(a+d)+(b+c)\\2(a-d)+(b-c)\\(a+d)-(b+c)\\(a-d)-2(b-c)\end{pmatrix}$$

$$\mathbf{C}_4^{\mathrm{T}}\begin{pmatrix}a\\b\\c\\d\end{pmatrix}=\begin{bmatrix}1&1&1&1/2\\1&1/2&-1&-1\\1&-1/2&-1&1\\1&-1&1&-1/2\end{bmatrix}\begin{pmatrix}a\\b\\c\\d\end{pmatrix}=\begin{pmatrix}a+b+c+d/2\\a+b/2-c-d\\a-b/2-c+d\\a-b+c-d/2\end{pmatrix}=\begin{pmatrix}(a+c)+(b+d/2)\\(a-c)+(b/2-d)\\(a-c)-(b/2-d)\\(a+c)-(b+d/2)\end{pmatrix}$$



1.1变换与量化

量化技术简介

- 量化（Quantization）是指将信号的连续取值（或大量的离散取值）映射为若干个离散取值的过程。视频残差信号经过变换之后，变换系数往往具有较大的动态范围，对变换系数进行量化可以有效减小信号取值空间，获得更好的（有损）压缩效果，但同时也因此成为视频编码**产生失真的根本原因**。
- H.264/AVC 视频压缩标准使用标量量化（Scalar Quantization）器。基本的前向量化器的操作是：

$$\mathbf{Z}_{ij} = \mathrm{round}\left(\frac{\mathbf{Y}_{ij} + f}{Q_{\mathrm{step}}}\right)$$

式中，$\mathbf{Y}_{ij}$是变换后的系数；Q_{step}是量化器的步长；f控制量化误差；$\mathbf{Z}_{ij}$是量化后的系数。

- H.264 标准建议量化帧内预测的变换系数时，$f = (2^{15} + \frac{QP}{6})/3$；量化帧间预测的变换系数时，$f = (2^{15} + \frac{QP}{6})/6$。这里的舍入（round）操作不需要舍入到最接近的整数，例如，有时向下取整反而有助于改善视频的感知质量。

1.1变换与量化

- H.264/AVC量化级别表

QP	Q_{step}	QP	Q_{step}	QP	Q_{step}	QP	Q_{step}	QP	Q_{step}
0	0.625	12	2.5	24	10	36	40	48	160
1	0.6875	13	2.75	25	11	37	44	49	176
2	0.8125	14	3.25	26	13	38	52	50	208
3	0.875	15	3.5	27	14	39	56	51	224
4	1	16	4	28	16	40	64		
5	1.125	17	4.5	29	18	41	72		
6	1.25	18	5	30	20	42	80		
7	1.375	19	5.5	31	22	43	88		
8	1.625	20	6.5	32	26	44	104		
9	1.75	21	7	33	28	45	112		
10	2	22	8	34	32	46	128		
11	2.25	23	9	35	36	47	144		

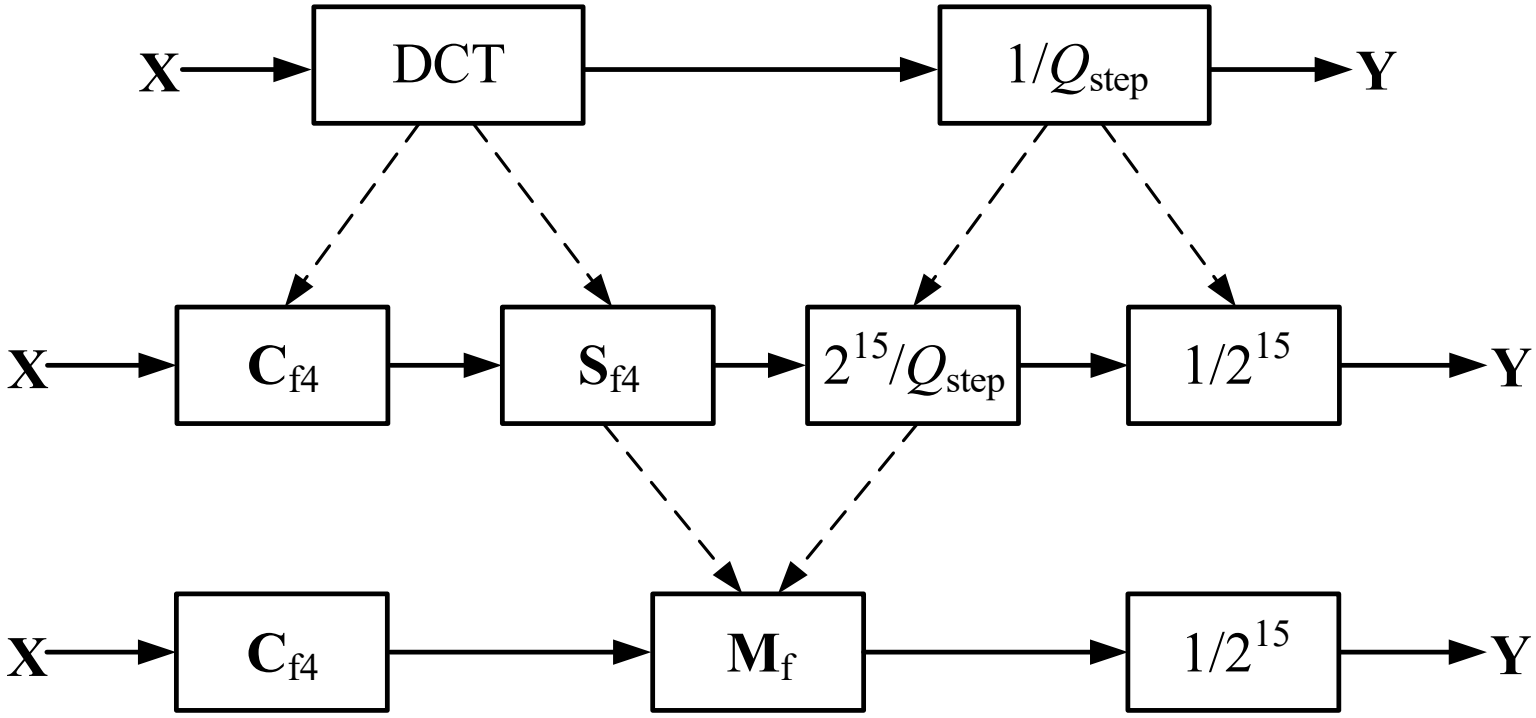
1.1变换与量化

- Q_{step}变化有明显规律：QP值每增加1，Q_{step}就增加12.25%；QP值增加6，Q_{step}就增加一倍。（此周期性特性有助于减少量化表和反量化表大小）

- 正向变换的⊗运算矩阵（记为$\mathbf{S}_f$）可以合并到量化表中，反向（逆）变换的⊗运算矩阵（记为$\mathbf{S}_i$）可以合并到反量化表中。

- 对于色度编码，一般使用和亮度编码相同的量化步长。H.264标准规定，亮度QP的最大值为51，色度QP的最大值为39。

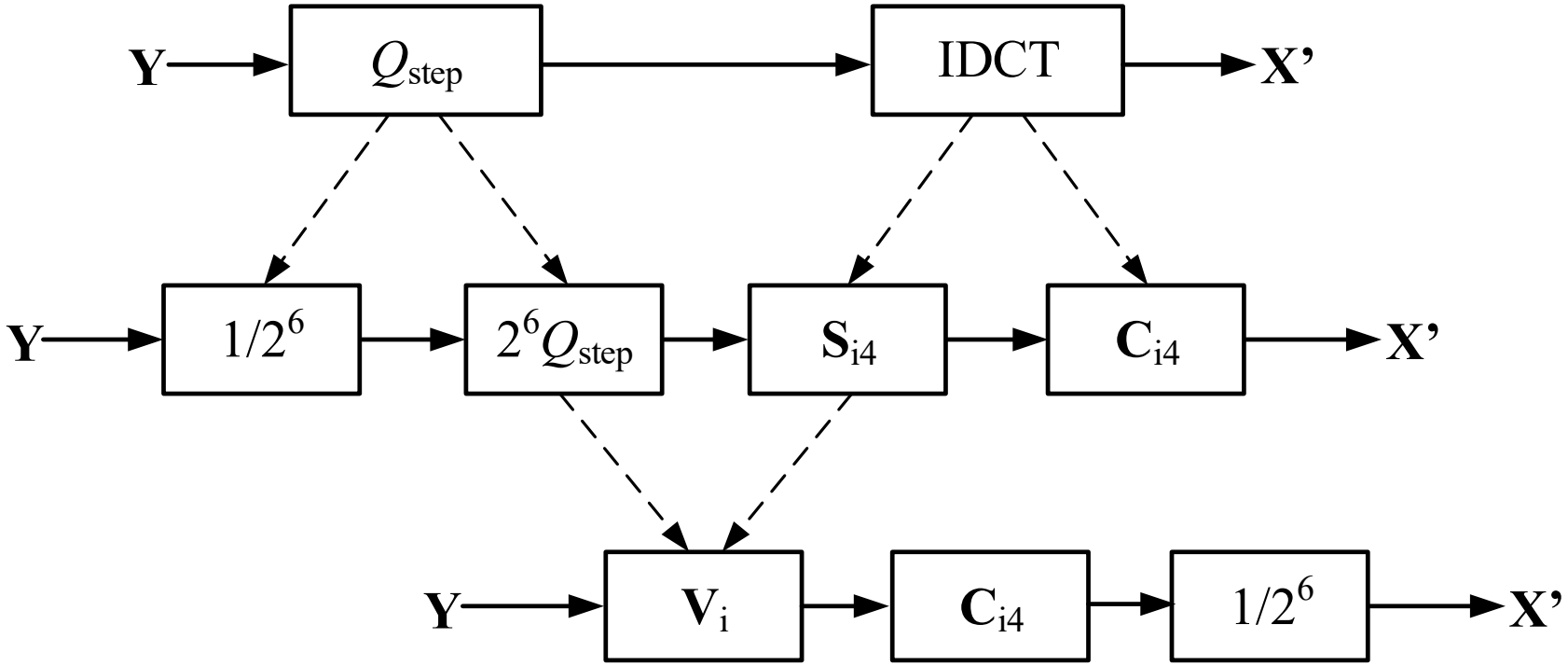
1.1变换与量化



$$\mathbf{M}_{\text{f}} = \frac{2^{15} \cdot \mathbf{S}_{\text{f4}}}{Q_{\text{step}}}$$

$$\mathbf{Y} = \text{round}\left(\left(\mathbf{C}_{\text{f4}} \mathbf{X} \mathbf{C}_{\text{f4}}^{\text{T}} \right) \otimes \mathbf{M}_{\text{f}} \cdot \frac{1}{2^{15}} \right)$$

1.1变换与量化



$$\mathbf{V}_{\mathrm{i}} = 2^6 \cdot Q_{\mathrm{step}} \cdot \mathbf{S}_{\mathrm{i4}}$$

$$\mathbf{X}' = \mathrm{round}\left(\mathbf{C}_{\mathrm{i4}} (\mathbf{Y} \otimes \mathbf{V}_{\mathrm{i}}) \mathbf{C}_{\mathrm{i4}}^{\mathrm{T}} \cdot \frac{1}{2^6} \right)$$

1.1变换与量化

$$\mathbf{M}_{\mathrm{f}}=\begin{bmatrix} m(QP,0) & m(QP,2) & m(QP,0) & m(QP,2) \\ m(QP,2) & m(QP,1) & m(QP,2) & m(QP,1) \\ m(QP,0) & m(QP,2) & m(QP,0) & m(QP,2) \\ m(QP,2) & m(QP,1) & m(QP,2) & m(QP,1) \end{bmatrix}$$

$$\mathbf{V}_{\mathrm{i}}=\begin{bmatrix} v(QP,0) & v(QP,2) & v(QP,0) & v(QP,2) \\ v(QP,2) & v(QP,1) & v(QP,2) & v(QP,1) \\ v(QP,0) & v(QP,2) & v(QP,0) & v(QP,2) \\ v(QP,2) & v(QP,1) & v(QP,2) & v(QP,1) \end{bmatrix}$$

QP	$m(QP,0)$	$m(QP,1)$	$m(QP,2)$	$v(QP,0)$	$v(QP,1)$	$v(QP,2)$
0	13107	5243	8066	10	16	13
1	11916	4660	7490	11	18	14
2	10082	4194	6554	13	20	16
3	9362	3647	5825	14	23	18
4	8192	3355	5243	16	25	20
5	7282	2893	4559	18	29	23
…	…	…	…	…	…	…

1.1变换与量化

- H.264/AVC整数离散余弦变换、量化流程示例

$$\mathbf{X}=\begin{bmatrix}57 & 60 & 55 & 62\\ 51 & 64 & 58 & 66\\ 64 & 61 & 62 & 62\\ 62 & 53 & 65 & 68\end{bmatrix}$$

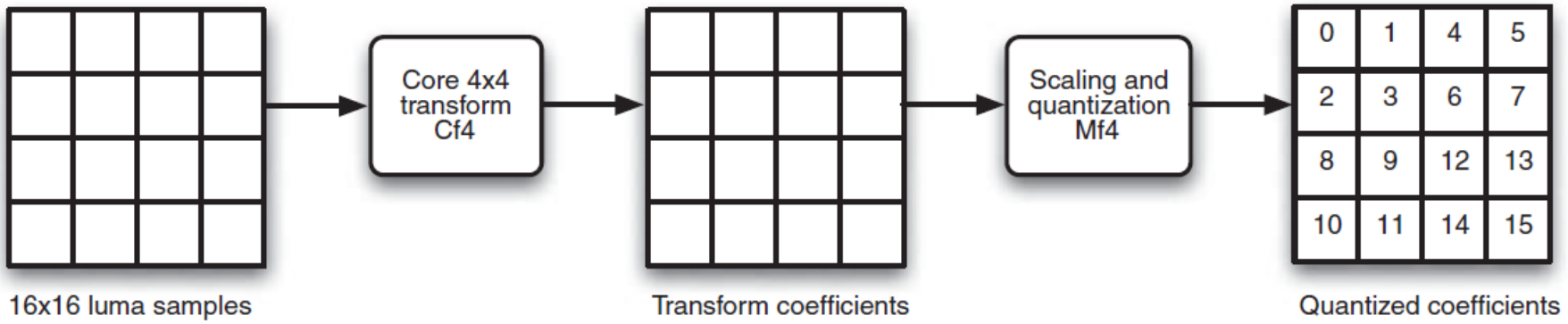
$$\mathbf{C}_{f4}\mathbf{X}\mathbf{C}_{f4}^{T}=\begin{bmatrix}970 & -50 & 14 & -20\\ -38 & 11 & -24 & -97\\ -6 & -8 & 18 & 26\\ 6 & 73 & 8 & 29\end{bmatrix}$$

$\mathbf{M}_f$ (QP = 30)

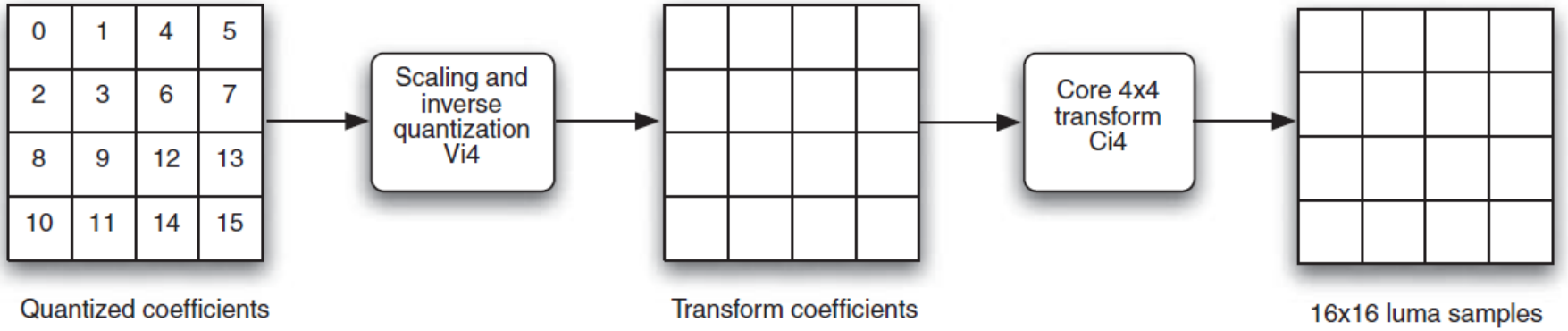
$$\begin{bmatrix}13107 & 8066 & 13107 & 8066\\ 8066 & 5243 & 8066 & 5243\\ 13107 & 8066 & 13107 & 8066\\ 8066 & 5243 & 8066 & 5243\end{bmatrix}/2^5$$

$$\mathbf{Y}=\mathrm{round}\left((\mathbf{C}_{f4}\mathbf{X}\mathbf{C}_{f4}^{T})\otimes\mathbf{M}_f\cdot\frac{1}{2^{15}}\right)=\begin{bmatrix}12 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}$$

1.1变换与量化

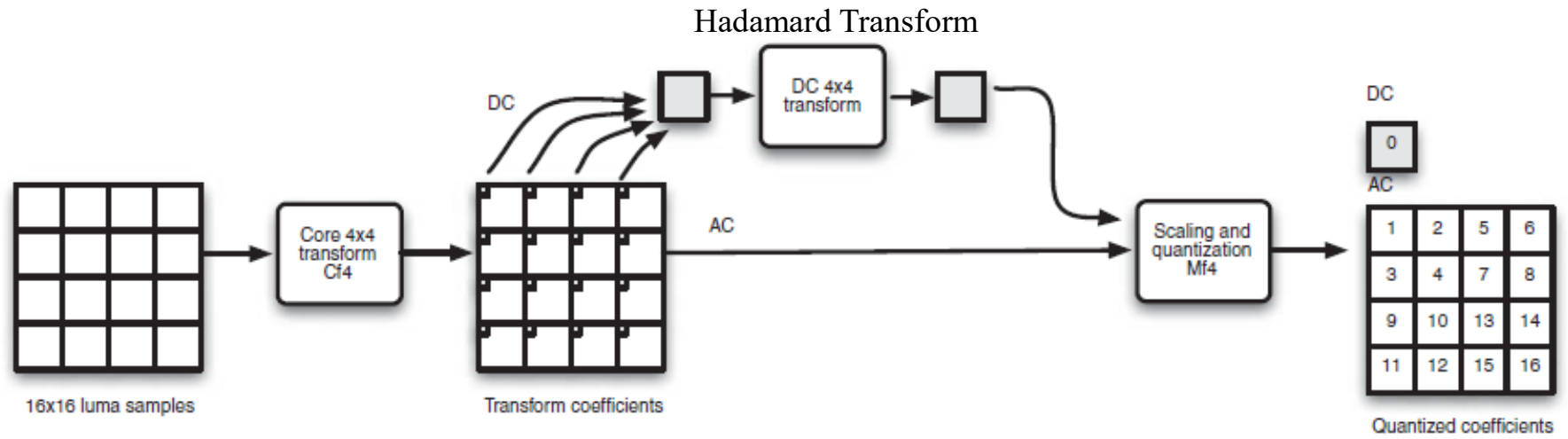


H.264/AVC亮度分量变换($\mathbf{C_{f4}}$)、量化($\mathbf{M_{f4}}$)整体流程（默认情况）

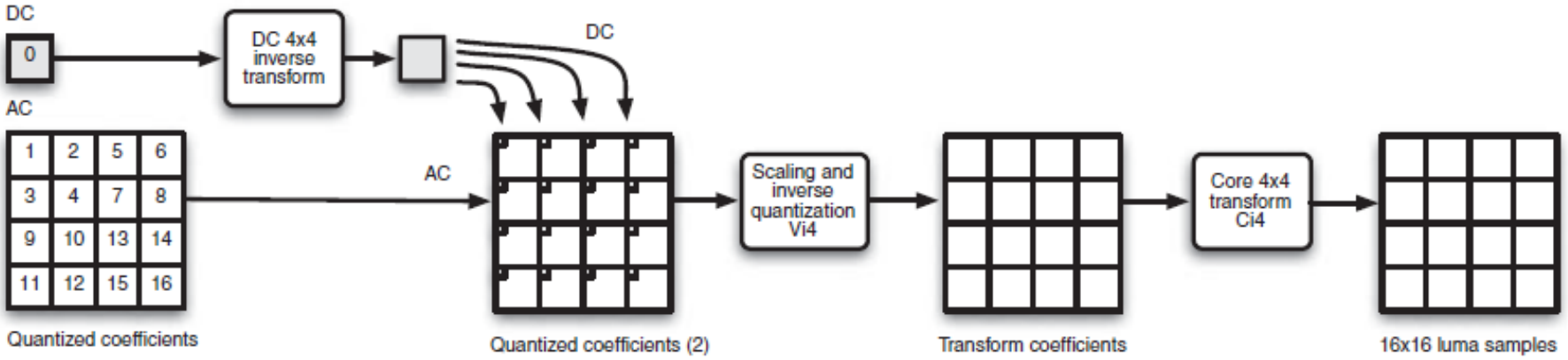


H.264/AVC亮度分量反量化($\mathbf{V_{i4}}$)、逆变换($\mathbf{C_{i4}}$)整体流程（默认情况）

1.1变换与量化

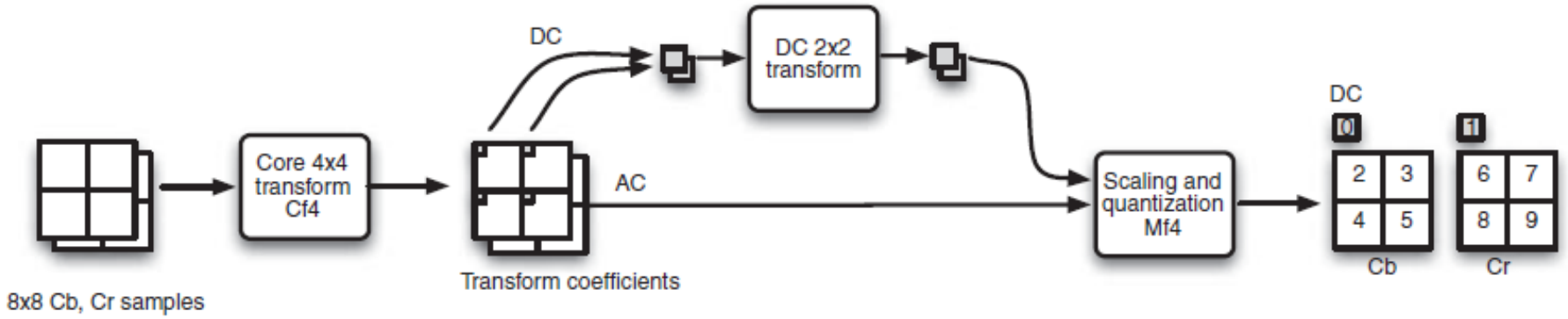


H.264/AVC亮度分量变换($\mathbf{C_{f4}}$)、量化($\mathbf{M_{f4}}$)整体流程（Intra_16x16）

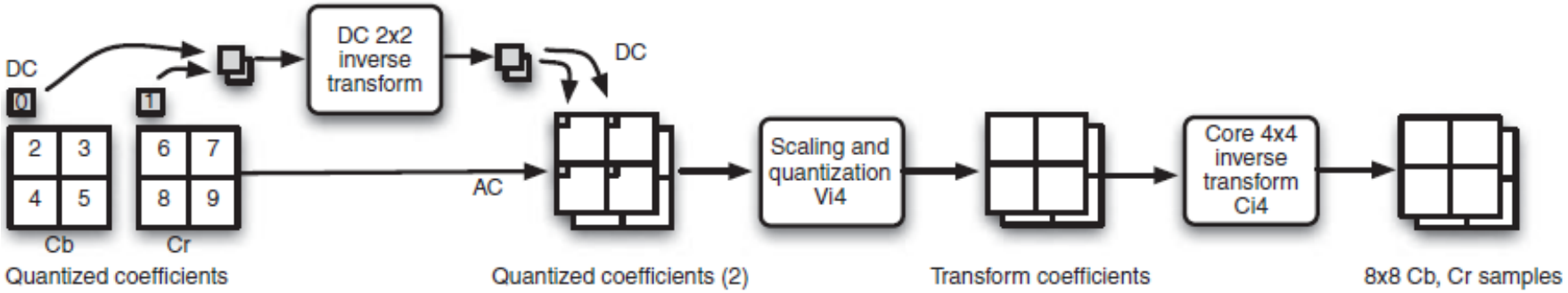


H.264/AVC亮度分量反量化($\mathbf{V_{i4}}$)、逆变换($\mathbf{C_{i4}}$)整体流程（Intra_16x16）

1.1变换与量化

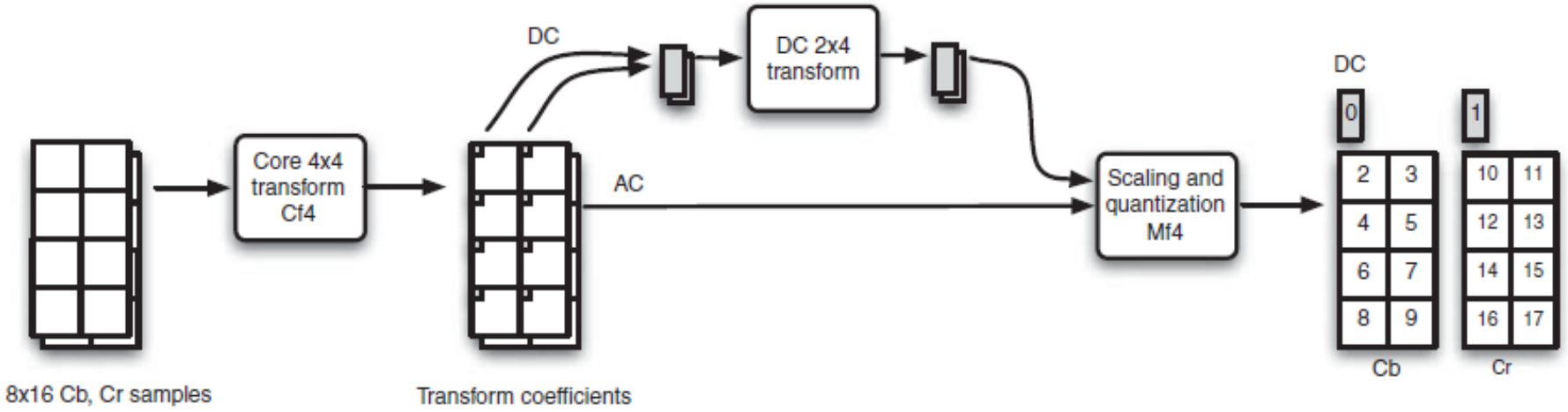


H.264/AVC色度分量变换($\mathbf{C_{f4}}$)、量化($\mathbf{M_{f4}}$)整体流程（YUV420）

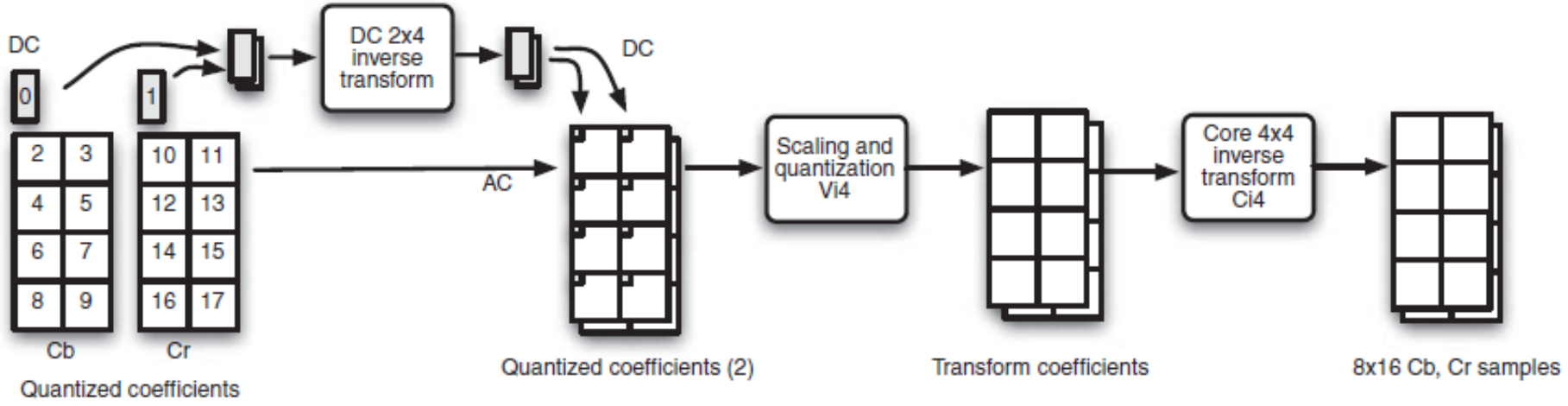


H.264/AVC色度分量反量化($\mathbf{V_{i4}}$)、逆变换($\mathbf{C_{i4}}$)整体流程（YUV420）

1.1变换与量化

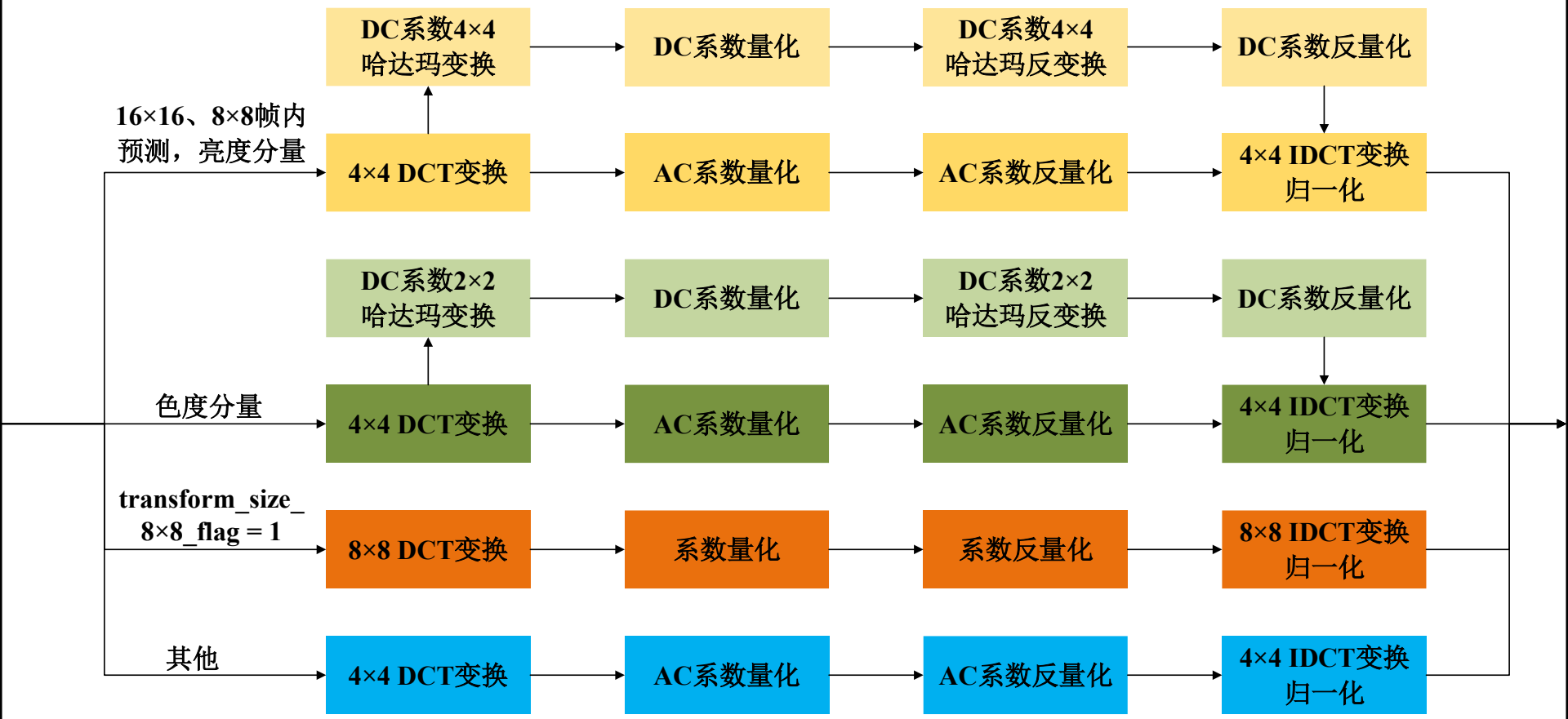


H.264/AVC色度分量变换($\mathbf{C_{f4}}$)、量化($\mathbf{M_{f4}}$)整体流程（YUV422）



H.264/AVC色度分量反量化($\mathbf{V_{i4}}$)、逆变换($\mathbf{C_{i4}}$)整体流程（YUV422）

1.1 变换与量化


DC系数4×4
哈达玛变换
DC系数量化
DC系数4×4
哈达玛反变换
DC系数反量化
16×16、8×8帧内
预测，亮度分量
4×4 DCT变换
AC系数量化
AC系数反量化
4×4 IDCT变换
归一化
DC系数2×2
哈达玛变换
DC系数量化
DC系数2×2
哈达玛反变换
DC系数反量化
色度分量
4×4 DCT变换
AC系数量化
AC系数反量化
4×4 IDCT变换
归一化
transform_size_
8×8_flag = 1
8×8 DCT变换
系数量化
系数反量化
8×8 IDCT变换
归一化
其他
4×4 DCT变换
AC系数量化
AC系数反量化
4×4 IDCT变换
归一化

1.2 熵编码

相关概念

- 消除信息熵冗余（编码冗余）
- 两种常用方案
  - ✓ 变长编码（Variable Length Coding，VLC）
    - ➢ 为概率大的符号分配短码字、为概率小的符号分配长码字
    - ➢ 例如：哈夫曼（Huffman）编码、指数哥伦布编码（Exp-Golomb Coding，EGC）
  - ✓ 算数编码（Arithmetic Coding）
    - ➢ 对输入符号序列进行操作
    - ➢ 将输入符号序列映射为实数轴上[0,1)区间内的一个小区间，区间宽度为符号序列的概率值，在此区间内选择一个二进制小数作为整个符号序列的编码码字
- 应用于视频编码的技术革新是在H.264中引入了上下文自适应技术。

1.2 熵编码

指数哥伦布编码

- 一种规则变长码，具有很好的结构性
- 由前缀和后缀两部分构成，均依赖于指数哥伦布码阶数K
- 第一个码字结构以1开头，后跟K个二进制位，即$1X_{K-1}X_{K-2}\cdots X_0$ $(X_i = 0,1)$；第二个码字结构以01开头，后跟（K+1）个二进制位；以此类推，每个码字结构都是前一个码字结构最左端添加一个0，最右端添加一个二进制位。

非负整数索引	码字					
	K=0		K=1		K=2	
	前缀	后缀	前缀	后缀	前缀	后缀
0	1	-	1	0	1	00
1	01	0	1	1	1	01
2	01	1	01	00	1	10
3	001	00	01	01	1	11
4	001	01	01	10	01	000
…	…	…	…	…	…	…

H.264/AVC采用的指数哥伦布编码

- 阶数$K=0$
- 码字结构：[ZERO PREFIX][1][INFO]
- 码字前缀：M个“0”后跟1个“1”，$M=\lfloor \log_2(\text{code\_num}+1) \rfloor$
- 码字后缀：M个二进制位，$\text{INFO}=\text{code\_num}-2^M+1$

code_num	Codeword
0	1
1	010
2	011
3	00100
4	00101
5	00110
6	00111
7	0001000
8	0001001
...	...

H.264/AVC采用的指数哥伦布编码

- 读取一系列“0”直到检测到“1”，确定“0”的数量M
- 读取第一个“1”后，接着读取长度为M的二进制序列，得到INFO
- $\text{code\_num} = \text{INFO} + 2^M - 1$

示例

- $\text{code\_num} = 106$

$M = \lfloor \log_2(106+1) \rfloor = 6$，故相应码字前缀为$0000001_2$

$\text{INFO} = 106 - 2^6 + 1 = 43 = 1010112$，其为相应码字后缀

拼接前缀、后缀得到码字为0000001101011

- 码字0000000111000011

检测到第一个“1”时，读取了7个连续“0”，故$M = 7$

读取第一个“1”后，接着读取长度为7的二进制序列，得到$\text{INFO} = 99_{10}$

$\text{code\_num} = 99 + 2^7 - 1 = 226$

H.264/AVC采用的指数哥伦布编码

- *ue* (unsigned direct mapping)

  无符号直接映射，用于编码宏块类型等。

  code_num $= k$

- *se* (signed mapping)

  符号映射，用于编码运动向量差值（difference）、量化参数差值等。

  $$\text{code\_num} = \begin{cases} 2|k| & k \le 0 \\ 2|k| - 1 & k > 0 \end{cases}$$

- *me* (mapped symbols)

  标记映射，用于编码coded_block_pattern（子宏块残差编码模式）

  code_num通过查找H.264标准中的相应表格（Table 9-4）得到

- *te* (truncated mapping)

  截断映射，取决于语法元素的取值范围，用于编码参考图像在参考帧列表中的序号等。

1.2 熵编码

上下文自适应变长编码

- Context Adaptive Variable Length Coding，CAVLC
- 用于亮度、色度残差数据编码，
- 与以往的单一码表的编码方式不同，CAVLC 通过多个码表切换逼近信源概率分布，以实现自适应编码，提高了熵编码的编码效率。
- 主要编码：系数标识（coeff_token，包括TotalCoeff和TrailingOnes）、拖尾系数符号、除拖尾系数外的非零系数幅值（Level）、最后一个非零系数前零的数量（TotalZeros）、每个非零系数前零的数量（RunBefore）
- 残差变换、量化后具有如下特性：
  - 4×4块残差经过变换量化后，非零系数主要集中在低频部分，高频系数大部分都是零
  - 直流系数附近的非零系数较大，高频非零系数不少为+1或-1
  - 非零系数的幅值变化有一定规律性和相关性，非零系数游程也具有一定特性
  - 相邻4×4块的非零系数的数量具有一定相关性

1.2 熵编码

CAVLC具体步骤

- 步骤1：编码非零系数数量（TotalCoeff）和拖尾系数数量（TrailingOnes）

  非零系数数量取值范围为0～16

  拖尾系数数量取值范围为0～3。拖尾系数是指经过zigzag扫描后，系数序列末尾连续出现的±1（中间可间隔任意数量0）。若±1的数量大于3，则只有最后3个被视为拖尾系数，其余被视为普通非零系数

  通过查表对TotalCoeff和TrailingOnes进行编码

- 步骤2：编码拖尾系数的符号

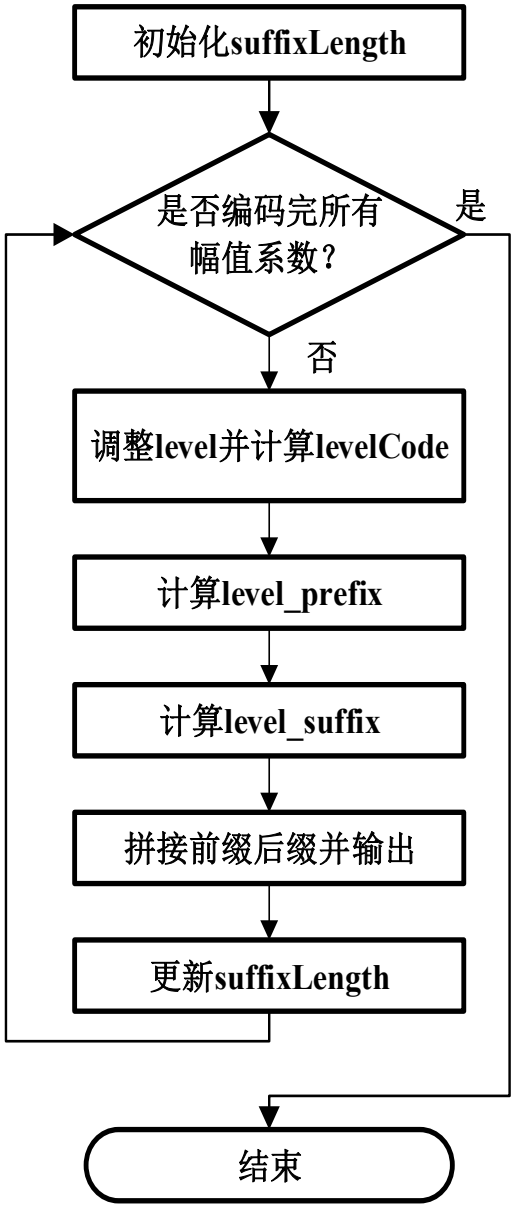
  按反向扫描的顺序，从高频数据开始，正号用“0”编码，负号用“1”编码。

- 步骤3：依次编码所有剩下的非零系数幅值（Level）

  根据编码系数Level进行自适应码表切换。包括两个部分：前缀（level_prefix）和后缀（level_suffix）。

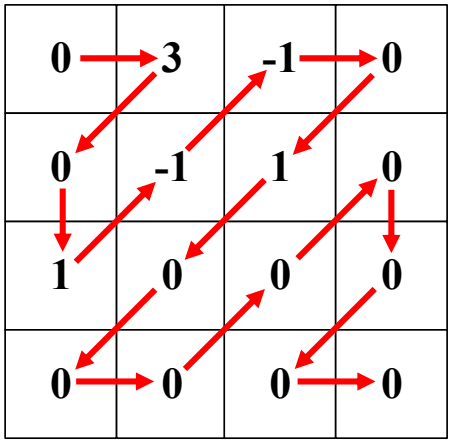
- 步骤4：编码最后一个非零系数前零的数量（TotalZeros）

- 步骤5：依次编码每个非零系数前零的数量（RunBefore）



CAVLC编码示例

- 经过zigzag扫描之后，数据重新排列为0，3，0，1，-1，-1，0，1，0…
- 初始值设定：非零系数的数目（TotalCoeff）＝5；拖尾系数的数目（TrailingOnes）＝3；最后一个非零系数前零的数目（TotalZeros）＝3；变量NC＝1；suffixLength＝0；i＝TotalCoeff＝5。
- 编码coeff_token：查H.264标准表9-5，根据TotalCoeff＝5，TrailingOnes＝3，有coeff_token＝0000 100。此时Code＝0000 100。
- 编码拖尾系数符号：逆序编码，三个拖尾系数的符号依次被编码为0，1，1。此时，Code＝0000 1000 11。



1.2 熵编码

CAVLC编码示例

- 编码除拖尾系数以外非零系数幅值：
  - $$\text{levelCode} = \begin{cases} (\text{level} \ll 1) - 2 & \text{level} > 0 \\ -(\text{level} \ll 1) - 1 & \text{level} < 0 \end{cases}$$
  - level_prefix = levelCode/(1 ≪ suffixLength)

查询H.264标准表9-6可得对应字符串的前缀

  - level_suffix = levelCode%(1 ≪ suffixLength)

level_suffix值的无符号二进制数形式为对应字符串的后缀

当前 suffixLength	阈　值
0	0
1	3
2	6
3	12
4	24
5	48
6	N

- 按照逆序，Level[i--] = 1；（此时i=1）levelCode = 0；level_prefix = 0；查H.264标准表9-6可得level_prefix = 0时对应的bit string = 1；因为suffixLength初始化为0，故该Level没有后缀；因为suffixLength = 0，故suffixLength++；此时，Code = 0000 1000 111。

- 编码下一个Level：Level[0] = 3；levelCode = 4；level_prefix = 2；查H.264标准表9-6可得bit string = 001；level_suffix = 0；suffixLength = 1；故码流为0010；此时，Code = 0000 1000 1110 010。

CAVLC编码示例

- 编码最后一个非零系数前零的数目（TotalZeros）：

  查H.264标准表9-7可得，当TotalCoeff = 5，total_zeros = 3时，bit string = 111。此时，Code = 0000 1000 1110 0101 11。

- 编码每个非零系数前零的数量（RunBefore）：i = TotalCoeff = 5；ZerosLeft = TotalZeros = 3；通过查H.264标准表9-10，按照逆序编码，有
  - ZeroLeft = 3，run_before = 1，run_before[4] = 10；
  - ZeroLeft = 2，run_before = 0，run_before[3] = 1；
  - ZeroLeft = 2，run_before = 0，run_before[4] = 1；
  - ZeroLeft = 2，run_before = 1，run_before[1] = 01；
  - ZeroLeft = 1，run_before = 1，run_before[0]不需要码流来表示。

- 最终，Code = 0000 1000 1110 0101 1110 1101。对该4 × 4块残差系数的编码完毕。

1.2 熵编码

上下文自适应二元算术编码

- Context-Adaptive Binary Arithmetic Coding，CABAC
- 以算数编码为核心，将其与一个设计精良的上下文模型相结合。
- 不同于VLC，算术编码能为符号分配非整数长度的码字，这样编码的长度可以更加接近于符号的熵，因此具有更高的编码性能。
- CABAC会为当前编码的语法元素选择合适的上下文模型，在此基础上，给出了当前语法元素与相关语法元素之间的条件概率，利用这个条件概率，可减少语法元素之间的冗余信息，提高压缩效率。
- CABAC为每个语法元素提供非静态的统计模型，使得编码器可根据编码内容自适应地调整语法元素的概率模型，进一步提高压缩效率。
- 与CAVLC相比，CABAC在编码效率上有9%～14%的性能提高，是目前应用于视频编码标准中编码效率最高的熵编码器之一。

1.3 去块效应环路滤波

块效应产生及其原因

- 块效应在视频编码中人眼可察觉到的小块边界处的不连续，只要是以块为单位进行视频编码（block-Based Video Coding）就会产生块效应。
- 分块DCT变换使得块与块之间的相关性被忽略。
- 不同分块分别进行量化处理，当相邻分块量化步长不同或舍弃的高频分量存在差异时，将破坏它们之间的相关性。
- 基于块的运动补偿（Motion Compensation）会加剧块效应。运动估计不会绝对准确。

1.3 去块效应环路滤波

去块效应滤波器（Deblocking Filter）

- 减轻、消除视频图像中的块效应
- 对块边界的像素滤波

后置滤波器（Post Filter）

- 只放置于解码端
- 处理位于解码环路外的显示缓冲区中的视频数据
- 后处理范畴，非标准化规范内容

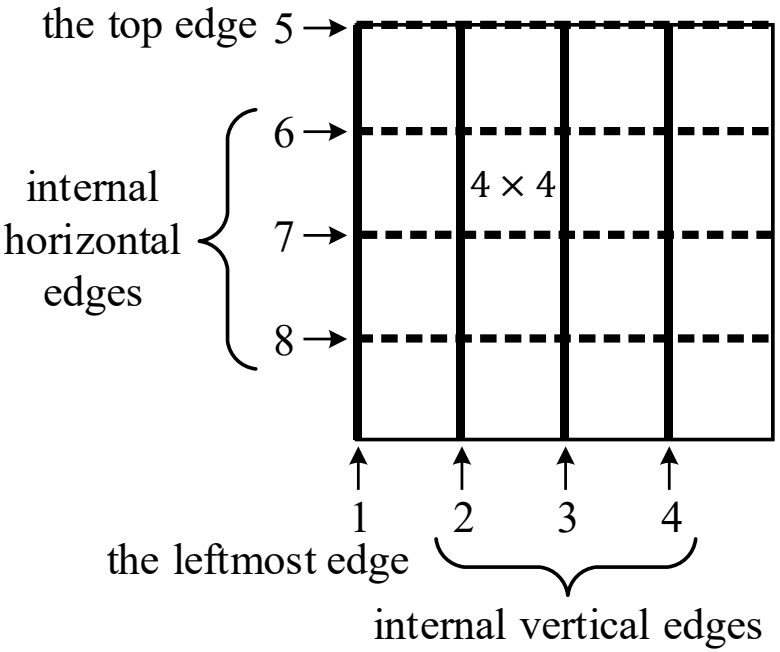
环路滤波器（In-Loop Filter）

- 放置于编解码环路中
- 编码器中：滤波图像作为帧间预测编码的（运动估计）参考图像
- 解码器中：滤波图像显示输出、作为解码重建的参考图像
- 编解码器使用相同滤波器，为标准化规范内容

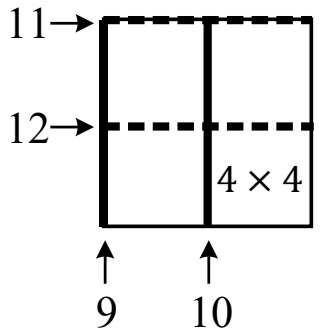
1.3 去块效应环路滤波

滤波次序

- 以宏块为单位
- 先横向后纵向，先亮度后色度，对4×4块边界进行滤波



宏块亮度分量滤波顺序



宏块色度分量滤波顺序（YUV420）

1.3 去块效应环路滤波

滤波过程

- 确定边界强度*BS* (Boundary Strength)

判断条件	*BS*
P块或Q块为帧内编码，且块边缘为宏块边界	4（最强滤波）
P块或Q块为帧内编码，且块边缘不为宏块边界	3
P块和Q块均非帧内编码，且残差变换系数不都为零	2
P块和Q块均非帧内编码，残差变换系数均为零，但它们的参考帧或运动向量不同	1
P块和Q块均非帧内编码，残差变换系数均为零，参考帧帧和运动向量均相同	0（无滤波）

- 色度块的边界强度无需计算，直接等于对应亮度块的边界强度

1.3 去块效应环路滤波

滤波过程

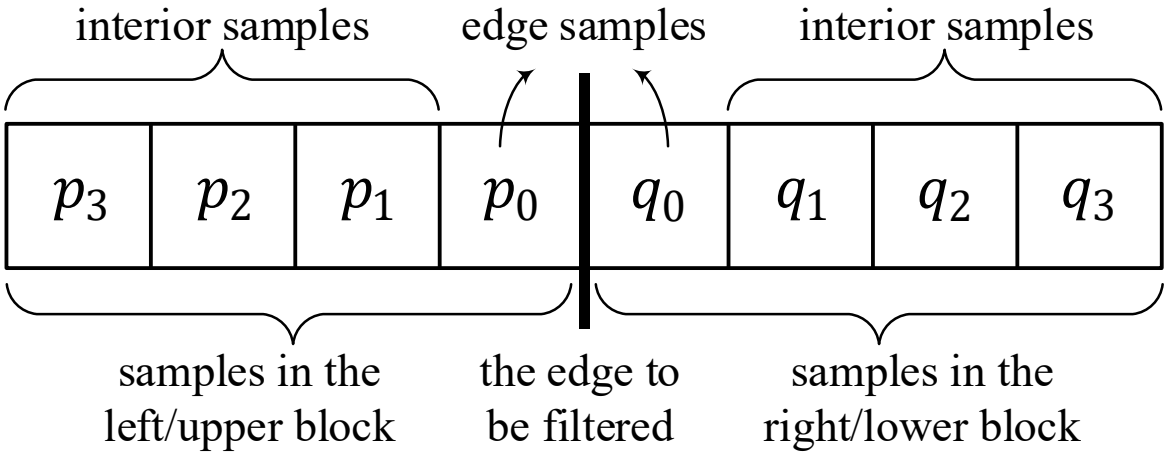
- 区分真假边界：当且仅当同时满足下述四个条件，则判定为需要滤波的虚假边界，否则判定为无需滤波的真实边界。其中，阈值α和β由相邻块的平均量化参数确定，且与其正相关。

$$Bs > 0$$

$$|p_0 - q_0| < \alpha \text{ (IndexA)}$$

$$|p_1 - p_0| < \beta \text{ (IndexB)}$$

$$|q_1 - q_0| < \beta \text{ (IndexB)}$$



1.3 去块效应环路滤波

IndexA和IndexB

$$\mathrm{IndexA} = \mathrm{Clip}(0,51, QP_{\mathrm{avg}} + \mathrm{FilterOffsetA})$$

$$\mathrm{IndexB} = \mathrm{Clip}(0,51, QP_{\mathrm{avg}} + \mathrm{FilterOffsetB})$$

偏移量FilterOffsetA和FilterOffsetB用于调整滤波强度，可由用户指定。

	IndexA for(α)或 IndexB for(β)																								
	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
α	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	5	6	7	8	9	10	12	13
β	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	2	3	3	3	3	4	4	4

	IndexA for(α)或 IndexB for(β)																								
	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
α	15	17	20	22	25	28	32	36	40	45	50	56	63	71	80	90	101	113	127	144	162	182	203	226	255	255
β	6	6	7	7	8	8	9	9	10	10	11	11	12	12	13	13	14	14	15	15	16	16	17	17	18	18

1.3 去块效应环路滤波

滤波运算（$Bs \in \{1,2,3\}$）

- 对边界像素p_0和q_0进行滤波：

  将p_1, p_0, q_0, q_1作为四拍（4-tap）线性滤波器的输入，产生滤波像素p_0'和q_0'

- 若$|p_2 - p_0| < \beta$，则对内部像素p_1进行滤波：

  将p_2, p_1, p_0, q_0作为四拍线性滤波器的输入，产生滤波像素p_1'

  （色度像素无需考虑此情况）

- 若$|q_2 - q_0| < \beta$，则对内部像素q_1进行滤波：

  将q_2, q_1, q_0, p_0作为四拍线性滤波器的输入，产生滤波像素q_1'

  （色度像素无需考虑此情况）

1.3 去块效应环路滤波

滤波运算（$Bs = 4$）

- 若$|p_2 - p_0| < \beta$且$|p_0 - q_0| < ((\alpha \gg 2) + 2)$，则对像素$p_2, p_1, p_0$进行滤波：

  将p_2, p_1, p_0, q_0, q_1作为五拍线性滤波器的输入，产生滤波像素p_0'

  将p_2, p_1, p_0, q_0作为四拍线性滤波器的输入，产生滤波像素p_1'

  将p_3, p_2, p_1, p_0, q_0作为五拍线性滤波器的输入，产生滤波像素p_2'（色度像素无需此操作）

- 若$|p_2 - p_0| \geq \beta$或$|p_0 - q_0| \geq ((\alpha \gg 2) + 2)$，则对像素$p_0$进行滤波：

  将p_1, p_0, q_1作为三拍线性滤波器的输入，产生滤波像素p_0'

- 对边界另一侧的像素q_i $(i \in \{0,1,2,3\})$进行上述类似操作

1.3 去块效应环路滤波

滤波效果实例（帧间编码）

原始视频帧

未滤波重建视频帧
QP=36

未滤波重建视频帧
QP=32

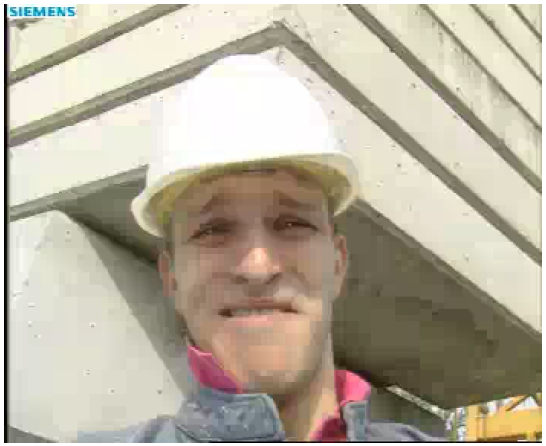
滤波重建视频帧
QP=36

滤波重建视频帧
QP=32

1.3 去块效应环路滤波

滤波效果实例（帧内编码）

原始视频帧



未滤波重建视频帧
QP=38

未滤波重建视频帧
QP=30

滤波重建视频帧
QP=38

滤波重建视频帧
QP=30

内 容 提 纲 （3 节 课 内 容）

1. H.264/AVC编码标准特性

2. H.265/HEVC简介

3. 小结

2 H.265/HEVC简介

什么是H.265/HEVC

- High Efficiency Video Coding，HEVC
- 一种视频压缩编码的国际标准

  由ISO/IEC的MPEG和ITU-T的VCEG这两个组织共同开发制定。
- 一种压缩视频格式

  标准规定了一种视频压缩编码格式及其相应的解析方法和工具。H.265/HEVC视频需要能够满足H.265标准所规定的码流规范，并且可以通过标准所提供的方法被正确解析。HEVC视频流可以存储于媒体文件，通过互联网进行点播、直播等。
- 一种视频编码工具集

  标准规定了一系列方法或工具，可供视频编码器采用。至于需要采用哪些工具、如何使用这些工具，由编码器设计者决定。

2 H.265/HEVC简介

为什么需要H.265/HEVC

- 数字视频的普及

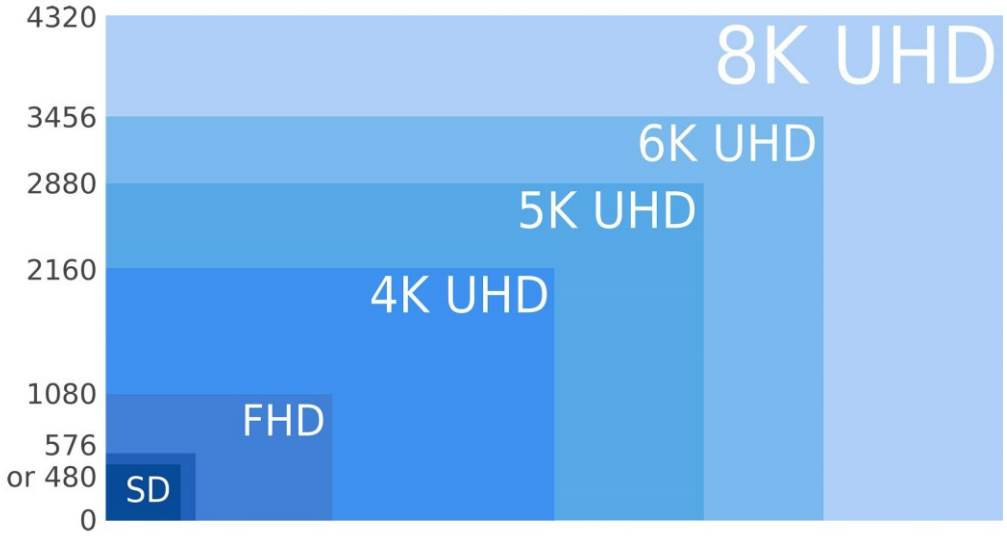
  视频点播、流媒体直播、视频会议、视频通话

- 视频解析度的增长

  480P → 720P → 1080P → 2k → 4k → 8k → …

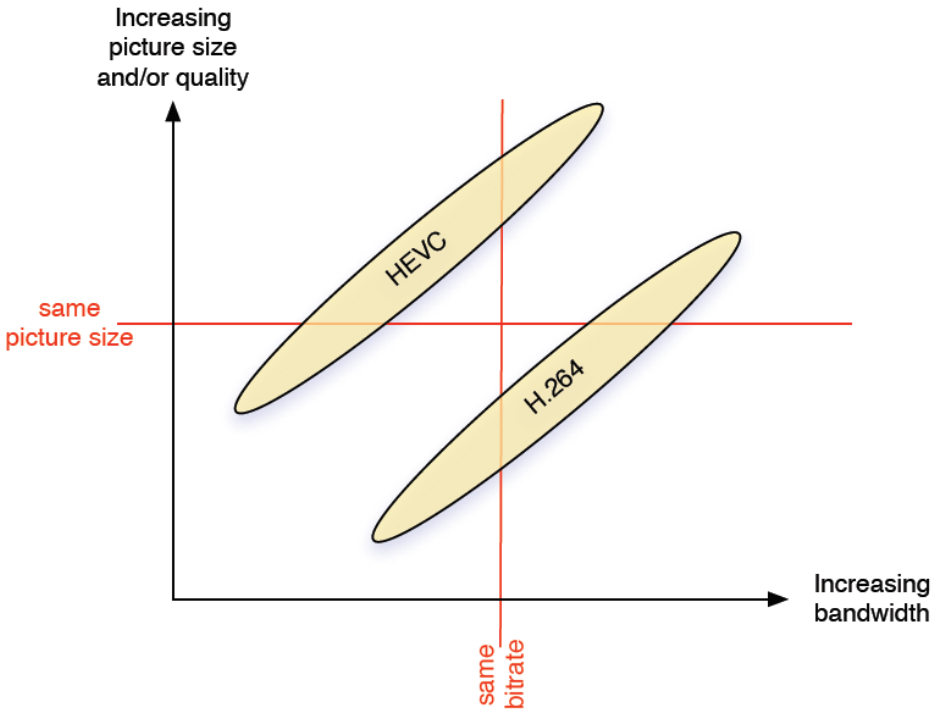
- 处理性能的提升

  当前移动终端的计算处理性能可超过10年前的普通家用计算机



为什么需要H.265/HEVC

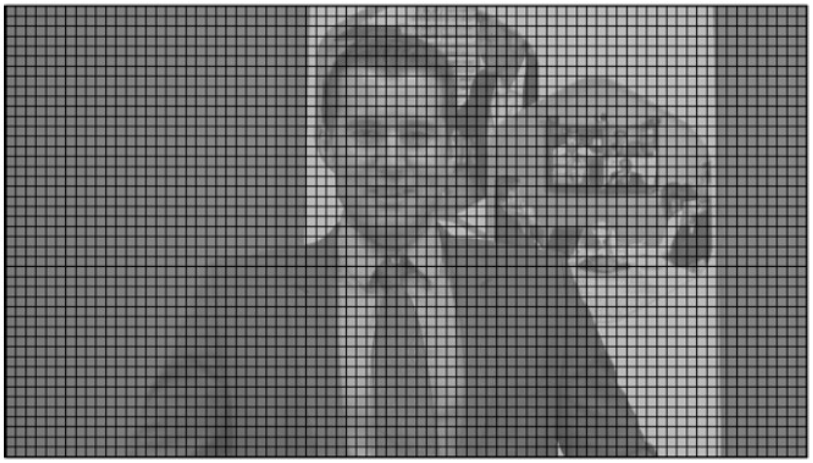
- 需要一种新的视频编码方案，能够充分利用计算资源，以应对海量高清数字视频的处理。
- H.265能够做到：
  - ✓在同等视频质量条件下，比H.264占用更少的存储空间和传输带宽
  - ✓在同等存储空间和传输带宽条件下，提供比H.264更高的视频质量



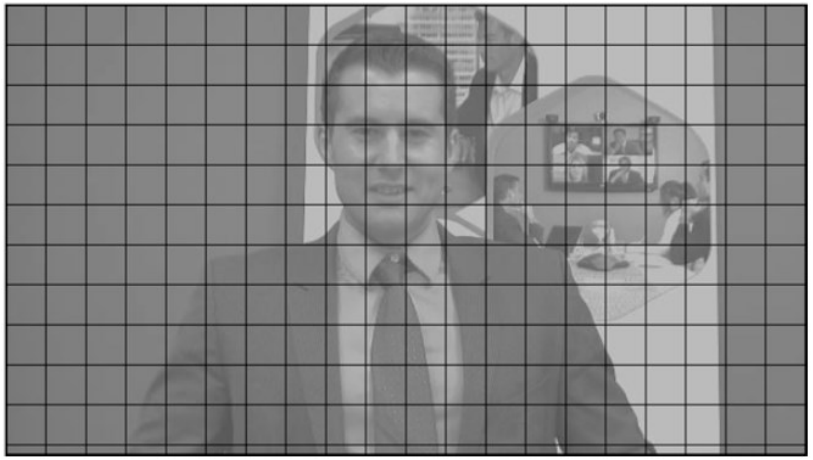
2 H.265/HEVC简介

更灵活的编码单元尺寸选择

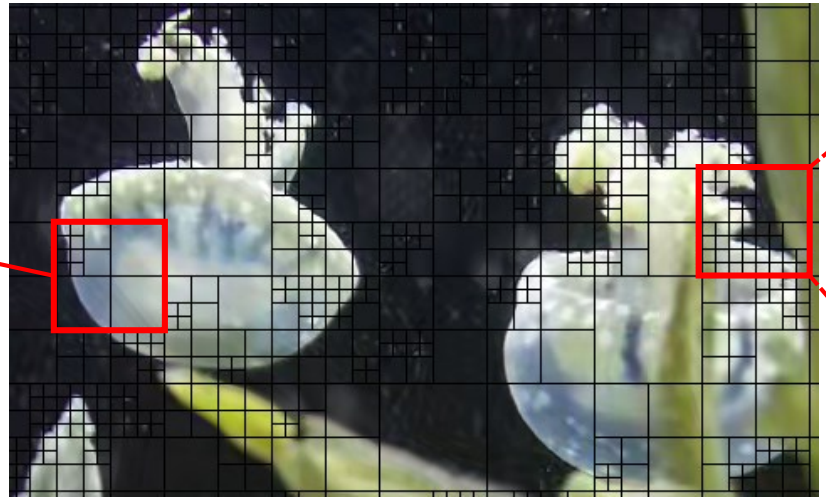
- 可将视频帧划分为64×64，32×32或16×16分块（CTU）

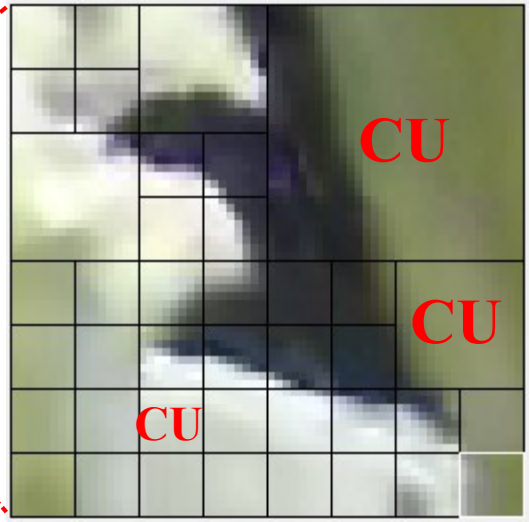


将16×16宏块作为编码单元



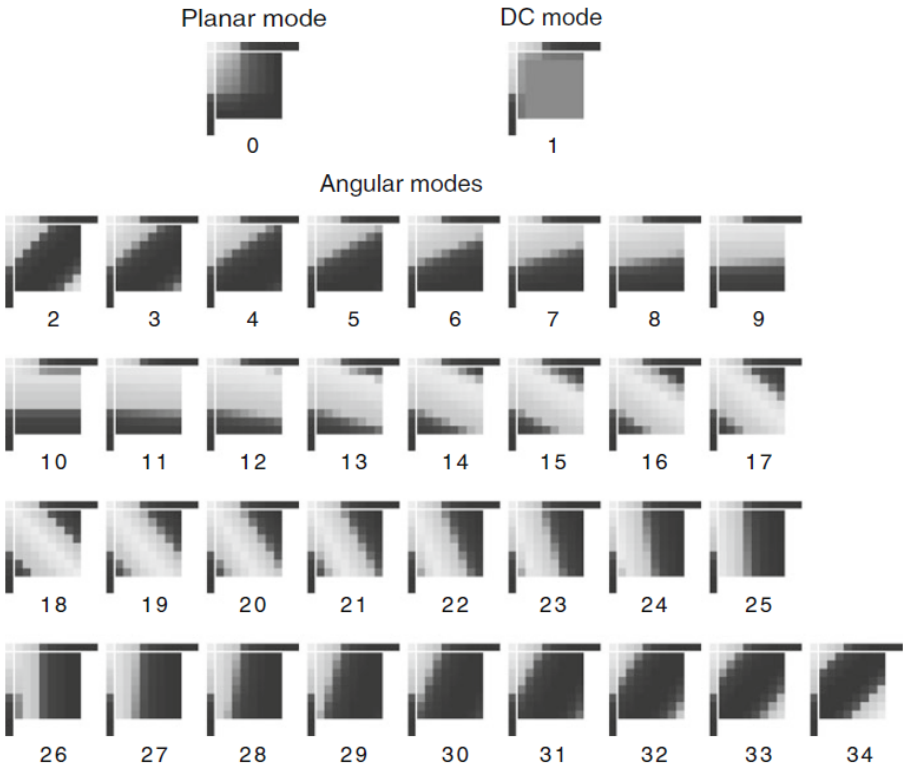
将64×64分块作为编码单元

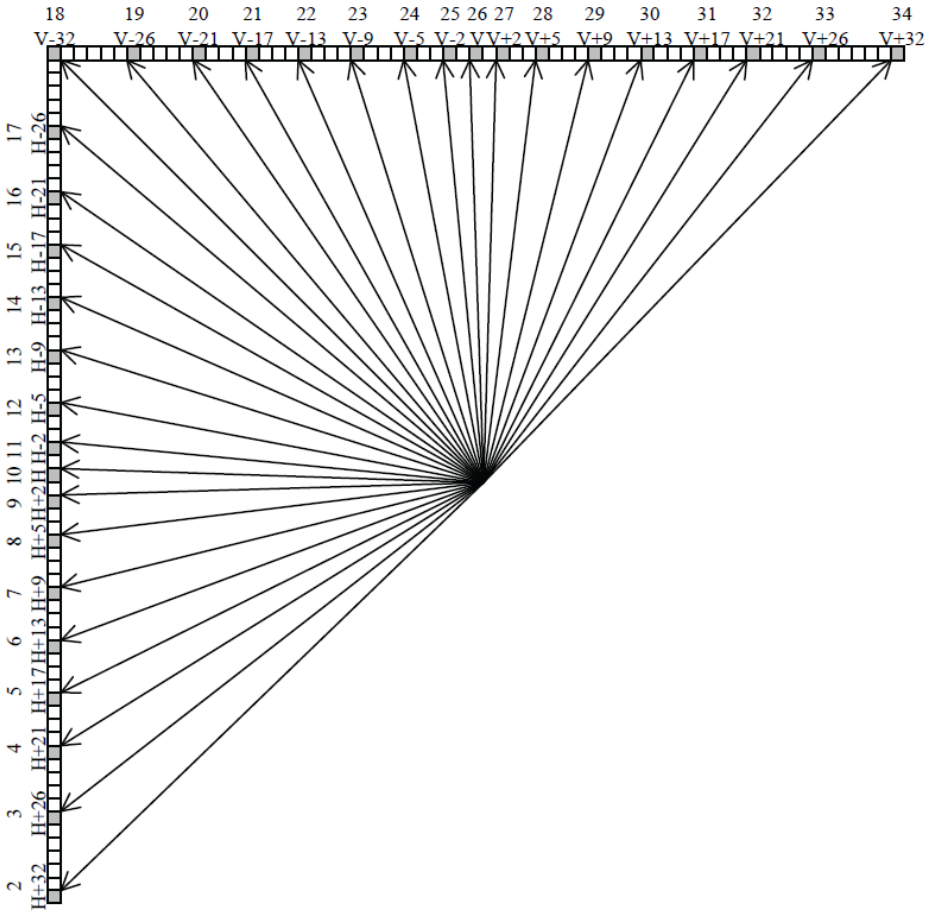




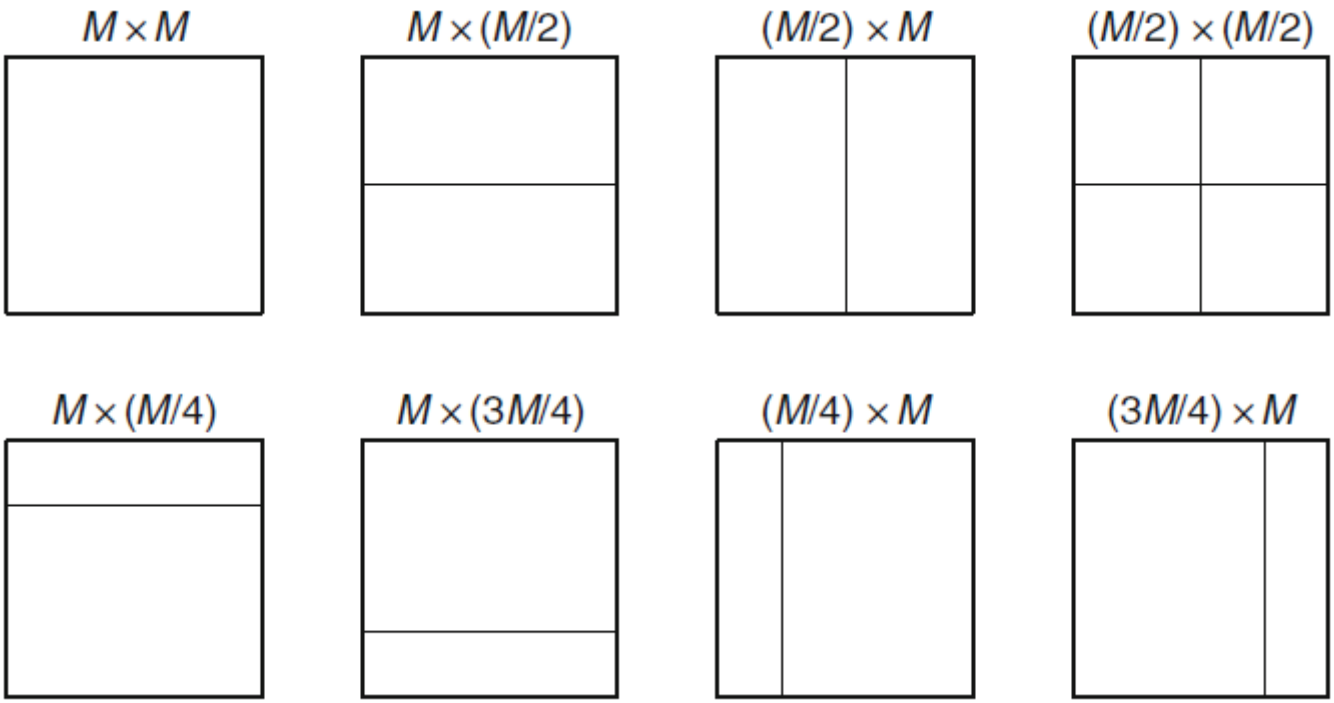
更丰富的帧内预测模式

- 35种帧内预测模式



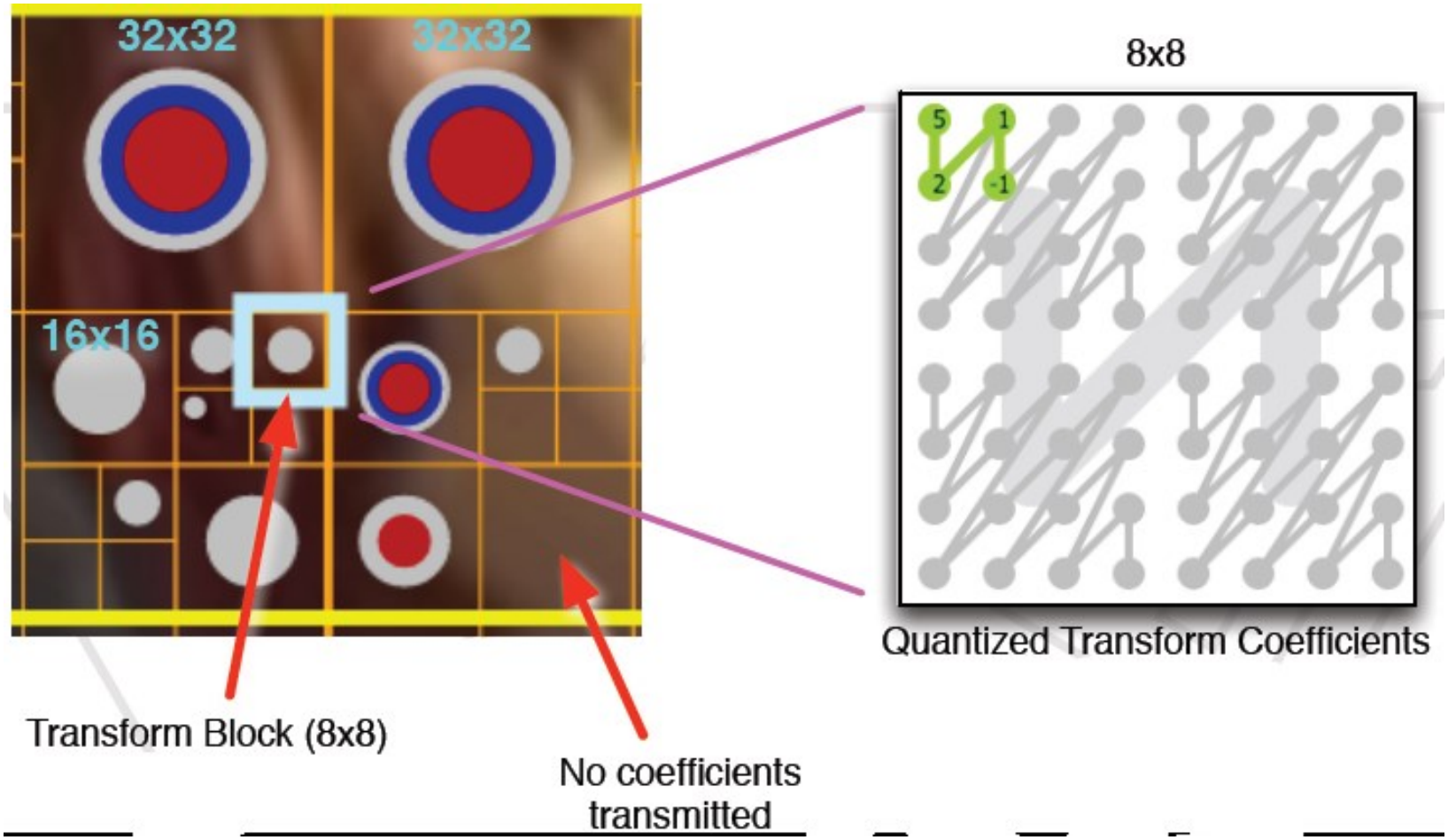


- 更灵活的帧间预测分块尺寸选择



更灵活的变换分块尺寸选择

- 变换分块尺寸可为4×4，8×8，16×16或32×32



内 容 提 纲 （3 节 课 内 容）

1. H.264/AVC编码标准特性

2. H.265/HEVC简介

3. 小结

3 小结

- H.264/AVC编码标准特性
  - 变换
  - 量化
  - 熵编码
  - 去块效应环路滤波
- H.265/HEVC简介

谢 谢

Q&A

欢迎电子邮件、QQ与微信交流问题！