Notes on H.264 Codec

Integer Transform and Quantization

Let X be a 4*4 matrix to be encoded, and Y the encoded data. The forward Integer Transform and quantization are done according to:

Y = round $([C_f]\times[X]\times[C_f^T]\cdot M_f/(2^{15}))$.

Here, the symbol "×" denotes normal matrix multiplication, while the symbol "·" denotes element-by-element multiplication. C_f is the 4*4 integer transform matrix:

$$\mathbf{C}_f = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}.$$

M_f is a 4*4 matrix for quantization generated form m, which is a 6*3 matrix (see Table I). For 0<QP<6, we have:

$$Mf = \begin{bmatrix} m(QP,0) & m(QP,2) & m(QP,0) & m(QP,2) \\ m(QP,2) & m(QP,1) & m(QP,2) & m(QP,1) \\ m(QP,0) & m(QP,2) & m(QP,0) & m(QP,2) \\ m(QP,2) & m(QP,1) & m(QP,2) & m(QP,1) \end{bmatrix}.$$

For QP >= 6, we replace each element $m(QP, n)$ in the above matrix with $m(QP\%6, \mathrm{n})/2^{\mathrm{floor}(QP/6)}$. By configuring QP, different quantization results can be achieved. The quantization is followed by a scaling step, which right shifts all quantized coefficients by 15 bits.

Inverse Integer Transform and De-Quantization

The inverse Integer Transform and de-quantization process is similar to its forward peer. Let Z be the decoded data, we have:

Z = round $([C_i^T]\times[Y\cdot V_i]\times[C_i]/(2^6))$.

Here Y is the encoded data obtained from the previous step, C_i is the inverse Integer Transform matrix:

$$\mathbf{C}_i = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix}.$$

V_i is the de-quantization table, which can be generated from v (see Table I). For $0<QP<6$, we have:

$$V_i = \begin{bmatrix} v(QP,0) & v(QP,2) & v(QP,0) & v(QP,2) \\ v(QP,2) & v(QP,1) & v(QP,2) & v(QP,1) \\ v(QP,0) & v(QP,2) & v(QP,0) & v(QP,2) \\ v(QP,2) & v(QP,1) & v(QP,2) & v(QP,1) \end{bmatrix}.$$

For $QP >= 6$, we replace each element $v(QP, n)$ in the above matrix with $v(QP\%6, n)*2^{\text{floor}(QP/6)}$. The de-quantization is also followed by a scaling step, which right shifts all restored value by 6 bits.

Table I: The value of m and v.

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

Note 1: Please refer to [1] for more information.

Note 2: The integer transform (forward and inverse) is separated into two 1-D transform by using the flow graphs described in [2]. Thereby no multiplication is needed (only addition and shift).

[1] White Paper: 4x4 Transform and Quantization in H.264/AVC, H264_4x4_transform_whitepaper_Nov10.pdf
[2] Henrique S. Malvar, et. al., Low-Complexity Transform and Quantization in H.264/AVC, IEEE Transactions on Circuits and Systems for Video Technology, 2003