



**White Paper:
Video Coding Walk-Through**

Iain Richardson / Vcodex

2011

Video coding walk-through

This example shows how a video frame is encoded (compressed) and decoded (decompressed). The frame is processed using a simple motion compensated transform encoder, similar to an early MPEG codec. However the basic principles apply to more sophisticated codecs such as H.264, VP8, etc.

The original frame is taken from a CIF video sequence, which means that the luma or luminance component shown in Fig. 1 is 352 samples wide and 288 samples high. The red/blue chroma components, not shown, are each 176x144 samples.

Fig. 2 shows the reconstructed previous frame in the video sequence. This has been encoded and decoded (=reconstructed) and some distortion can be seen. The difference between Fig 1 and Fig 2 **without** motion compensation is shown in Fig. 3. There is clearly still significant energy in Fig. 3, especially around moving areas.



Fig 1. Input frame



Fig 2. Reconstructed reference frame



Fig. 3 Residual : no motion compensation

Motion estimation is carried out with a 16x16 block size and half-pixel accuracy, producing the set of vectors shown in Fig. 4, superimposed on the current frame for clarity. Many of the vectors are zero and are shown as dots, which means that the best match for the macroblock is in the same position in the reference frame. Around

moving areas, the vectors point in the direction that blocks have moved **from**. The man standing on the left of the picture is walking to the left; the vectors therefore point to the **right**, i.e. where he has come from. Some of the vectors do not appear to correspond to “real” movement, for example those on the surface of the table, but indicate simply that the best match is not at the same position in the reference frame. “Noisy” vectors like these often occur in regions of the picture where there are no clear object features in the reference frame.



Fig. 4 16x16 motion vectors superimposed on frame

Fig. 5 shows the motion compensated reference frame, i.e. the reference frame with each block shifted according to the motion vectors. The walking person has been moved to the left to provide a better match for the same person in the current frame and the hand of the left-most person has been moved down to provide an improved match. Subtracting the motion compensated reference frame from the current frame gives the motion-compensated residual (Fig. 6). Compared with Fig. 3, the energy has clearly been reduced, particularly around moving areas.



Fig. 5 Motion compensated reference frame

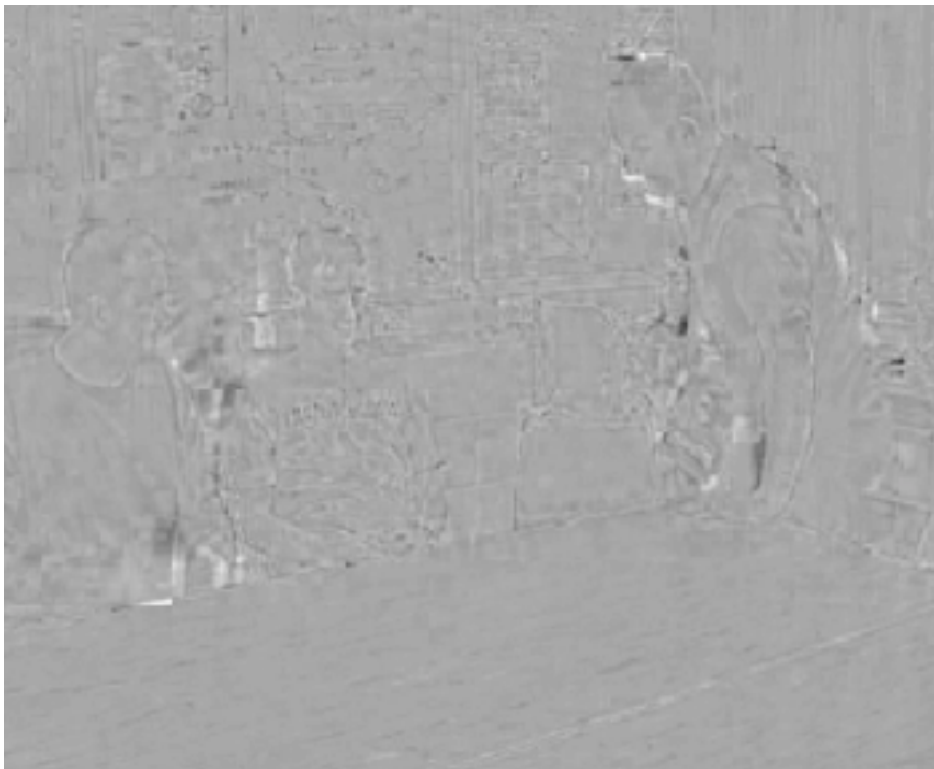


Fig. 6 Motion compensated residual frame

Fig. 7 shows a macroblock from the original frame, taken from around the head of the figure on the right. Fig. 8 is the luma component of the same macroblock after motion compensation. This is known as the residual.



Fig. 7 Original macroblock : luminance

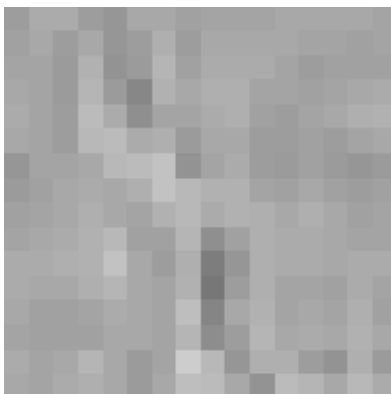


Fig. 8 Residual macroblock : luminance

Each block within the residual macroblock is transformed using a Discrete Cosine Transform (DCT). Applying a two-dimensional DCT to the top-right 8x8 block of luminance samples (Table 1) produces the DCT coefficients listed in Table 2. The magnitude of each coefficient is shown in Fig. 9. Note that the larger coefficients are clustered around the top-left or DC coefficient.

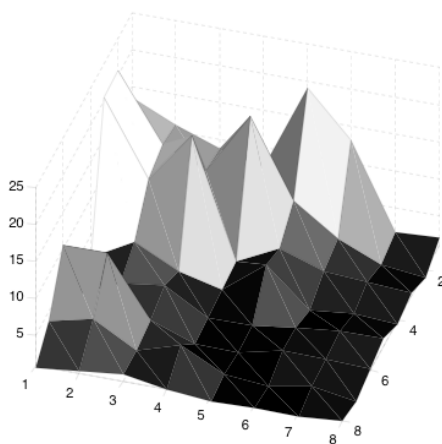


Fig. 9 DCT coefficient magnitudes : top-right 8x8 block)

Table 1 Residual luminance samples : top-right 8x8 block

-4	-4	-1	0	1	1	0	-2
1	2	3	2	-1	-3	-6	-3
6	6	4	-4	-9	-5	-6	-5
10	8	-1	-4	-6	-1	2	4
7	9	-5	-9	-3	0	8	13
0	3	-9	-12	-8	-9	-4	1
-1	4	-9	-13	-8	-16	-18	-13
14	13	-1	-6	3	-5	-12	-7

Table 2 DCT coefficients

-13.50	20.47	20.20	2.14	-0.50	-10.48	-3.50	-0.62
10.93	-11.58	-10.29	-5.17	-2.96	10.44	4.96	-1.26
-8.75	9.22	-17.19	2.26	3.83	-2.45	1.77	1.89
-7.10	-17.54	1.24	-0.91	0.47	-0.37	-3.55	0.88
19.00	-7.20	4.08	5.31	0.50	0.18	-0.61	0.40
-13.06	3.12	-2.04	-0.17	-1.19	1.57	-0.08	-0.51
1.73	-0.69	1.77	0.78	-1.86	1.47	1.19	0.42
-1.99	-0.05	1.24	-0.48	-1.86	-1.17	-0.21	0.92

A forward quantizer is applied:

$$Q_{coeff} = \text{round}(\text{coeff}/Q_{step})$$

where Q_{step} is the quantizer step size, $Q_{step}=12$ in this example. The output of the quantizer is shown in Table 3. Small-valued coefficients become zero in the quantized block and the non-zero outputs are clustered around the top-left (DC) coefficient.

Table 3 Quantized coefficients

-1	2	2	0	0	-1	0	0
1	-1	-1	0	0	1	0	0
-1	1	-1	0	0	0	0	0
-1	-1	0	0	0	0	0	0
2	-1	0	0	0	0	0	0
-1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

The quantized block is re-ordered in a zigzag scan starting at the top-left to produce the following list of coefficients:

-1, 2, 1, -1, -1, 2, 0, -1, 1, -1, 2, -1, -1, 0, 0, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 1, 0,... (all zeros after this)

This array is processed to produce a series of (run, level) pairs:

(0, -1)(0,2)(0,1)(0,-1)(0,-1)(0,2)(1,-1)(0,1)(0,-1)(0,2)(0,-1)(0,-1)(2,-1)(3,-1)(0,-1)(5,1)(EOB)

The first number in each pair indicates the number of preceding zeros and the second number indicates the non-zero "level". "EOB" (End Of Block) indicates that the

remainder of the coefficients are zero.

Each (run, level) pair is encoded as a variable length code (VLC). Using the variable length code tables from MPEG-4 Visual, the VLCs shown in Table 4 are produced.

Table 4 Variable length coding example

Run, Level, Last	VLC including sign bit
(0,-1, 0)	101
(0,2, 0)	11100
(0,1, 0)	100
(0,-1, 0)	101
(0,-1, 0)	101
(0,2, 0)	11100
(1,-1, 0)	1101
...	...
(5,1, 1)	00100110

The final VLC signals that LAST=1, indicating that this is the end of the block. The motion vector for this macroblock is (0, 1), i.e. the vector points downwards. The motion vector X and Y components are (differentially) coded as (1) and (0010) respectively.

The macroblock is transmitted as a series of VLCs, including a macroblock header, coded motion vector and transform coefficients for each 8x8 block.

At the decoder, the VLC sequence is decoded to extract header parameters, coded motion vectors and (run,level) pairs for each block. The 64-element array of coefficients is reconstructed by inserting (run) zeros before every (level). The array is then ordered to produce an 8x8 block identical to Table 3. The quantized coefficients are rescaled using:

$$R_{coeff} = Q_{step} \cdot Q_{coeff}$$

Rescaling or inverse quantization produces the block of coefficients shown in Table 5. This block is significantly different from the original DCT coefficients (Table 2) due to the quantization process. An Inverse DCT is applied to create a decoded residual block (Table 6) which is similar but not identical to the original residual block (Table 1). Comparing the original and decoded residual blocks in Fig. 10, it is clear that the decoded block has less high-frequency variation because of the loss of high-frequency DCT coefficients through quantization.

Table 5 Rescaled coefficients

-12	24	24	0	0	-12	0	0
12	-12	-12	0	0	12	0	0
-12	12	-12	0	0	0	0	0
-12	-12	0	0	0	0	0	0
24	-12	0	0	0	0	0	0
-12	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Table 6 Decoded residual luminance samples

-3	-3	-1	1	-1	-1	-1	-3
5	3	2	0	-3	-4	-5	-6
9	6	1	-3	-5	-6	-5	-4
9	8	1	-4	-1	1	4	10
7	8	-1	-6	-1	2	5	14
2	3	-8	-15	-11	-11	-11	-2
2	5	-7	-17	-13	-16	-20	-11
12	16	3	-6	-1	-6	-11	-3

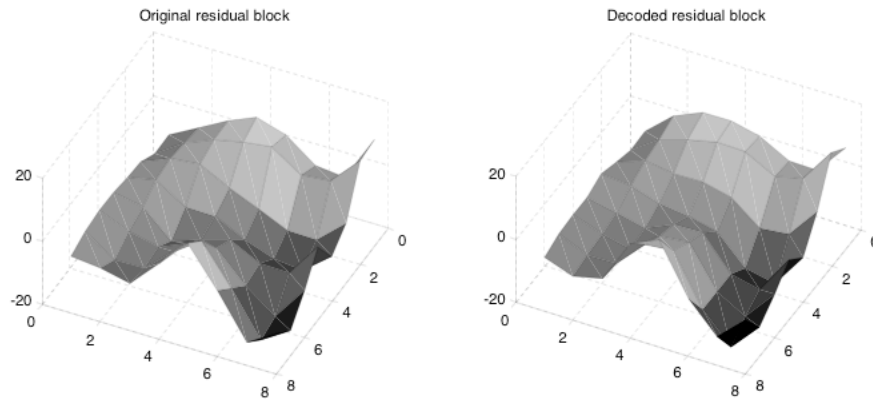


Fig. 10 Original and decoded residual blocks

The decoder recreates the original motion vector (0, 1). Using this vector, together with its own copy of the previously decoded frame (Fig. 2), the decoder reconstructs the macroblock. The complete decoded frame is shown in Fig. 11. Because of the quantization process, some distortion has been introduced, for example around detailed areas such as the faces and the writing on the board and there are some obvious edges along 8x8 block boundaries. The complete sequence was compressed by around 300 times, i.e. the coded sequence occupies less than 1/300 the size of the uncompressed video. Significant compression is achieved at the expense of relatively poor image quality.



Fig. 11 Decoded frame

1.1 Summary

Further reading

Iain E Richardson, "The H.264 Advanced Video Compression Standard", John Wiley & Sons, 2010.

About the author

Iain Richardson wrote the books on H.264 video compression : see <http://vcodex.com/h264book/>. A founder of OneCodec, he is changing the way video coding works.

Iain Richardson
iain@onecodec.com
<http://onecodec.com>