INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC1/SC29/WG11
MPEG 98/M3991
October 1998

Source: University of California at Berkeley
Purpose: Proposal
Title: Matching Pursuit Coding for Fine Granular Video Scalability
Authors: Sen-ching Samson Cheung, Avideh Zakhor[1]

This document describes the use of Matching Pursuit (MP) coding in providing fine granular video scalability and some preliminary results. The motivation and the test conditions for this core experiment can be found in document m3881.

Algorithm

The structure of a simple fine granular scalable coder with guaranteed base layer decoding is shown below:

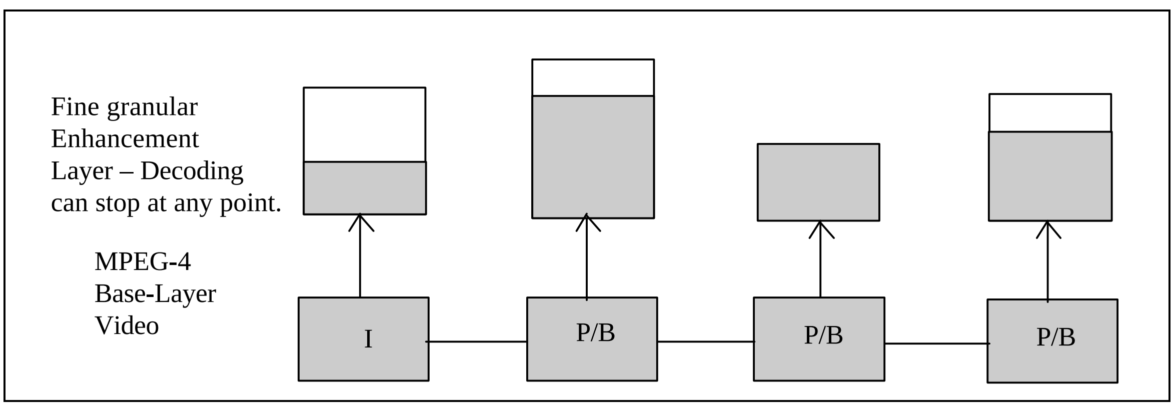

Figure 1: Simple Scalability Structure (grey area - bits decoded by a particular decoder)

As a particular decoder can decode any portion of the available enhancement layer, the architecture shown in Figure 1 does not exploit any temporal dependency.

MP decomposes a signal progressively into *atoms* which are chosen from an over-complete library of functions based on their contributions towards the signal. MP was originally proposed for inter-texture coding in MPEG4 Visual standard version 2. Being a greedy algorithm, MP is also suitable for fine granular scalable coding. Due to the similarity in the signal characteristics between the enhancement texture and the inter-texture, we have adopted the same atom search strategy and the same library as described in VM 11 (N2172.)

[1] For more information on this document, please contact Dr. Avideh Zakhor at avz@eecs.berkeley.edu.

For MP inter-texture coding, atoms are coded in a particular order of their spatial locations. If the same order is used for enhancement layer coding, the results will be sub-optimal as significant atoms (those have large contributions to the signal energy) may scatter all over the picture. To achieve the proper order of significance and at the same time exploit spatial correlation, we introduce the concept of a *packet*. Packets are groups of atoms and they are ordered in terms of their significance. Within a single packet, atoms are arranged in terms of their spatial positions. The following figure demonstrates this idea:

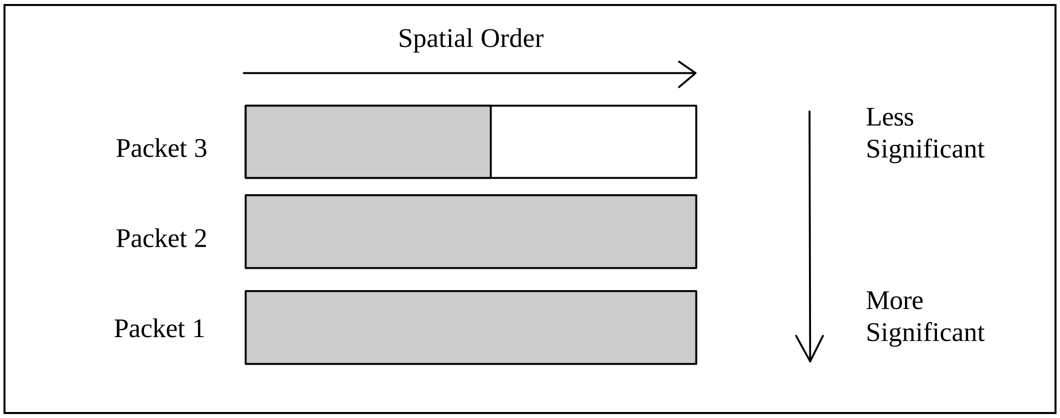

Figure 2 : Packets of atoms (gray area - bits decoded by a particular decoder)

It turns out that the atom coding method described in VM 11, which is macroblock based, is rather inefficient if too few atoms are present in a packet. An alternative method, which codes the atoms in raster scan order of their global positions, is thus used. The details of this coding scheme and the variable length code tables used are described in an earlier version of the VM (VM 8, N1796.) Despite the fact that atoms within a packet are not arranged in the order of significance, our experiments show that it is still beneficial to decode part of a packet to meet the target bitrate exactly. This is probably due to the fact that most of the basis functions are localized in space and thus the "reordering" error only occurs at the small area around the position where the decoding stops.

The format of a packet is described as follows:

	No. of bits	Mnemonic
numatoms	16	uimsbf
if (first packet) **vop_mp_quant**	2	uimsbf
vert_skip	9	uimsbf
first_atom_in_line = 1;		
for (i=0; i<**numatoms**; i++)		
mpatom(i, first_atom_in_line)		

mpatom(i, first_atom_in_line) {		
if (first_atom_in_line)		
first_atom_in_line = 0		
position_p1	5-18	vlclbf
} else {		
position_p2	1-18	vlclbf
if (**position_p2** == "position_escape")		
first_atom_in_line = 1		
position_p3	1-16	vlclbf
position_p1	5-18	vlclbf

}		
horizontal_shape	3-7	vlclbf
if (**horizontal_shape** == “Cb_escape”) || (**horzontal_shape**== “Cr_shape”)		vlclbf
horizontal_shape	3-7	
vertical_shape	3-6	vlclbf
modulus_sign	1	bslbf
modulus_code	1-18	vlclbf
if (**modulus_code** == “Escape”)		
modulus_flc	8	uimsbf
}		

The semantics and the VLC tables for the following set of keywords are described in VM 11 (N2172):

vop_mp_quant Number of refinements of quantization for atom modulus.
horizontal_shape Horizontal component index of an atom.
vertical_shape Vertical component index of an atom.
modulus_sign Sign of the modulus.
modulus_code Magnitude of the modulus.
modulus_flc Fixed length code for the modulus.

The semantics and the VLC tables of the following set of keywords are described in VM 8 (N1796):

numatoms Number of atoms in the current packet.
vert_skip Number of lines from the top of the picture to the first atom.
position_p1 The horizontal distance of the first atom on the line.
position_p2 The differential horizontal distance of the atom from the previous on the same line.
position_p3 The differential vertical distance of the atom from the previous.

It should be noted that the MP decoder relies on external information to determine the actual size of a packet.

Results

We have applied MP coding to the three sequences (Coastguard and Foreman in QCIF and CIF, Stefan in SIF) as described in the test condition document (m3881). The coherence of our source sequences, our base layer coding, as well as the three required decoder bit traces were all cross-verified with all the participating parties of the core experiment.

For simplicity, we have used constant number of atoms per packet. For all the QCIF sequences, each packet has 50 atoms. For all the CIF sequences, each packet has 100 atoms and for the SIF sequence, each packet has 500 atoms. These numbers represent a trade-off between coding efficiency and quality. Larger packets are coded more efficiently because the position VLC tables have shorter codewords for smaller differential distances. On the other hand, it is difficult to meet the target rate precisely with integral numbers of large packets. Thus large portion of bits are spent for the last partial packet which is sub-optimal. For each simulation, we have run two sets of experiments - one with a mid-processing step which consists of the VM’s deblocking and deringing filters and the other without.

Table 1 shows the gains in PSNR of the enhancement layer over the base layer for each simulation. Despite the small differences in PSNR between the set with mid-processing and the one without, the visual differences are quite visible. The set using the mid-processing has most of the base layer artifacts removed and the MP coding is able to allocate atoms to true features and produce better rendering of details. Finally, the numbers of bits wasted correspond to at most one atom which is very small (maximum 40 bits in Table 1.) The PSNR plots for the set with mid-processing are shown in figures 3 to 7.

Table 1: PSNR Gain from Base Layer for various sequences under various decoding traces.
(mid = midprocessing; Trace 0 = staircase; Trace 1 = ramps; Trace 2 = flat)

Decoding Traces	Y-PSNR (dB)	U-PSNR (dB)	V-PSNR (dB)	Max # of bits wasted
Coastguard QCIF				
Base	27.40	38.78	41.69	-
Trace 0	+2.17	+1.13	+1.14	34
Trace 0 (mid)	+2.30	+1.42	+1.26	28
Trace 1	+1.45	+0.43	+0.90	29
Trace 1 (mid)	+1.51	+0.67	+0.98	32
Trace 2	+2.53	+1.37	+1.32	33
Trace 2 (mid)	+2.69	+1.61	+1.43	32
Foreman QCIF				
Base	28.15	35.29	34.48	-
Trace 0	+2.74	+2.47	+3.18	39
Trace 1 (mid)	+3.17	+2.84	+3.80	30
Trace 1	+1.89	+1.67	+2.32	32
Trace 1 (mid)	+2.24	+2.05	+2.88	35
Trace 2	+3.10	+2.76	+3.76	31
Trace 2 (mid)	+3.57	+3.19	+4.42	31
Coastguard CIF				
Base	26.52	37.66	40.95	-
Trace 0	+2.24	+1.71	+0.86	33
Trace 0 (mid)	+2.29	+1.84	+1.01	28
Trace 1	+1.57	+0.62	+0.68	31
Trace 1 (mid)	+1.57	+0.77	+0.84	40
Trace 2	+2.67	+2.02	+1.15	29
Trace 2 (mid)	+2.75	+2.16	+1.30	35
Foreman CIF				
Base	29.22	34.94	35.43	-
Trace 0	+2.40	+2.43	+3.00	37
Trace 0 (mid)	+2.62	+2.83	+3.52	38
Trace 1	+1.74	+1.70	+2.12	35
Trace 1 (mid)	+1.91	+2.06	+2.62	36
Trace 2	+2.76	+2.79	+3.40	34
Trace 2 (mid)	+3.01	+3.19	+3.96	29
Stefan SIF				
Base	28.37	34.15	33.79	-
Trace 0	+2.94	+2.87	+3.04	28
Trace 0 (mid)	+3.12	+3.13	+3.27	27
Trace 1	+2.07	+2.02	+2.18	28
Trace 1 (mid)	+2.18	+2.22	+2.36	32
Trace 2	+3.54	+3.45	+3.63	32
Trace 2 (mid)	+3.76	+3.76	+3.91	31

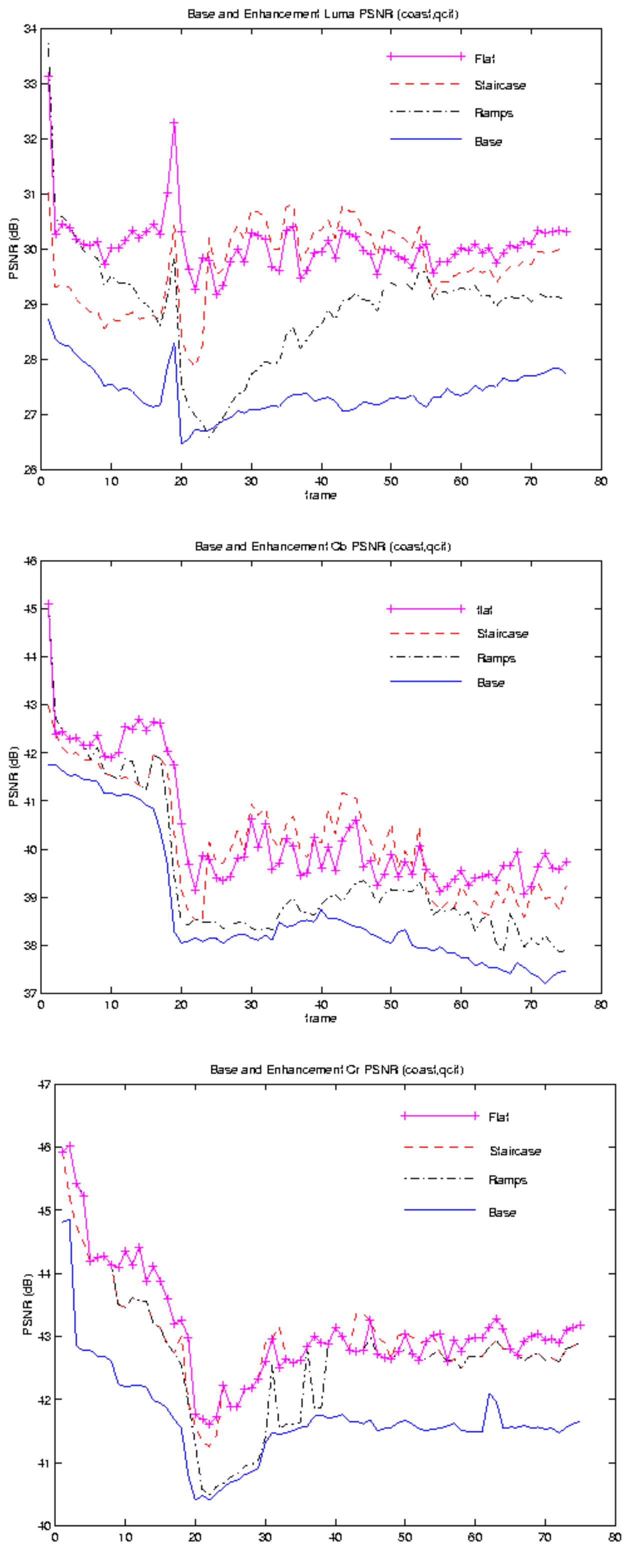

Figure 3: PSNR plots for Coastguard QCIF (with mid-processing)

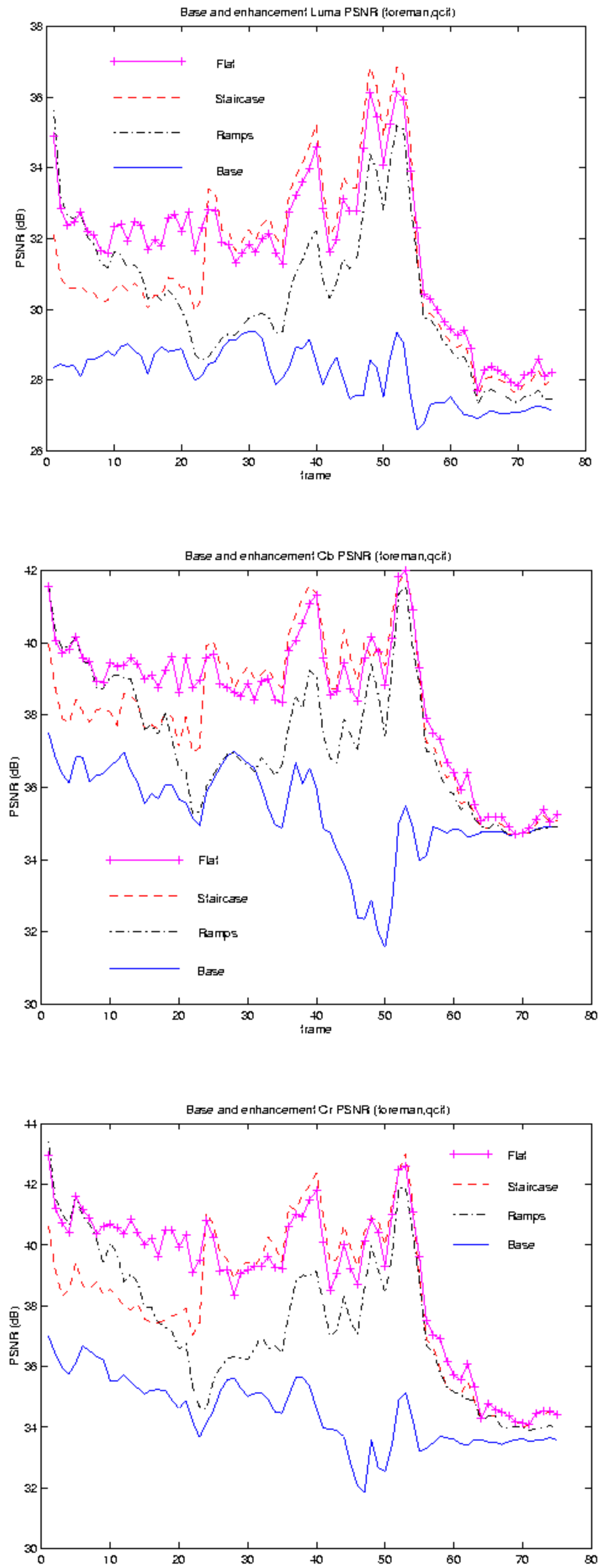

Figure 4 PSNR plots for Foreman, QCIF (with mid-processing)

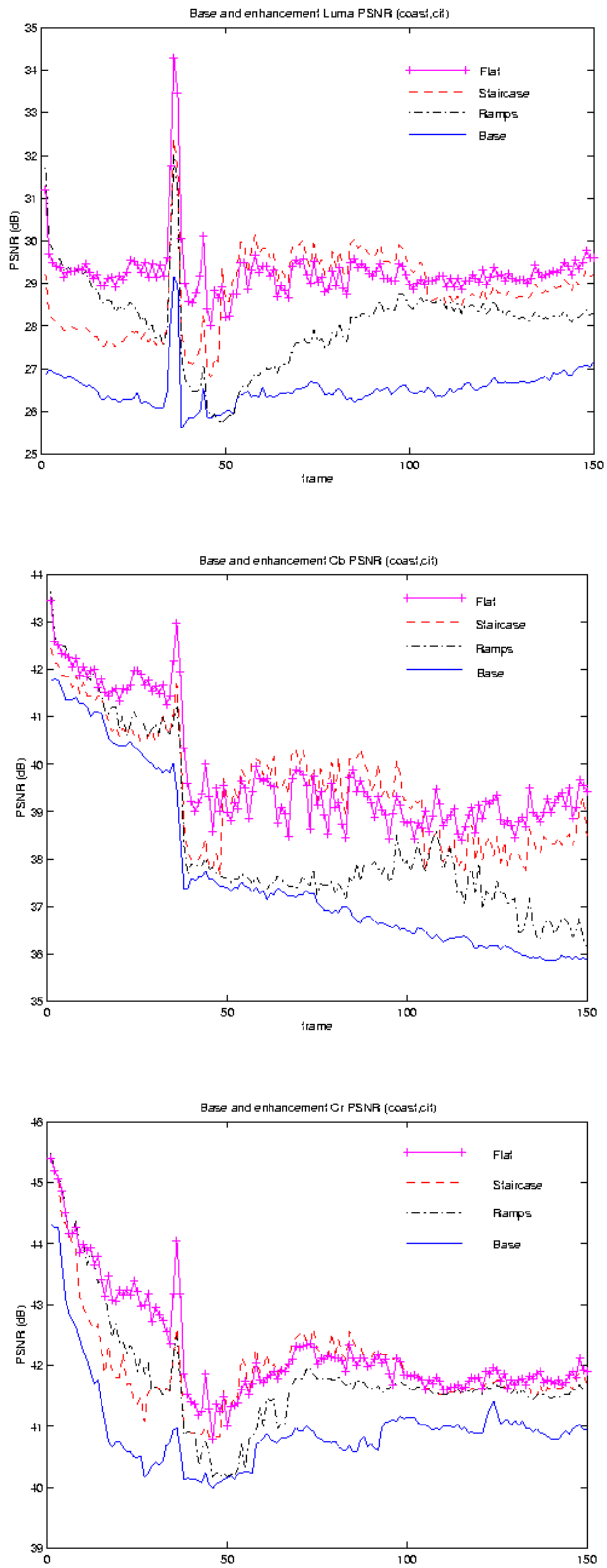

Figure 5 PSNR plots for Coastguard, CIF (with mid-processing)

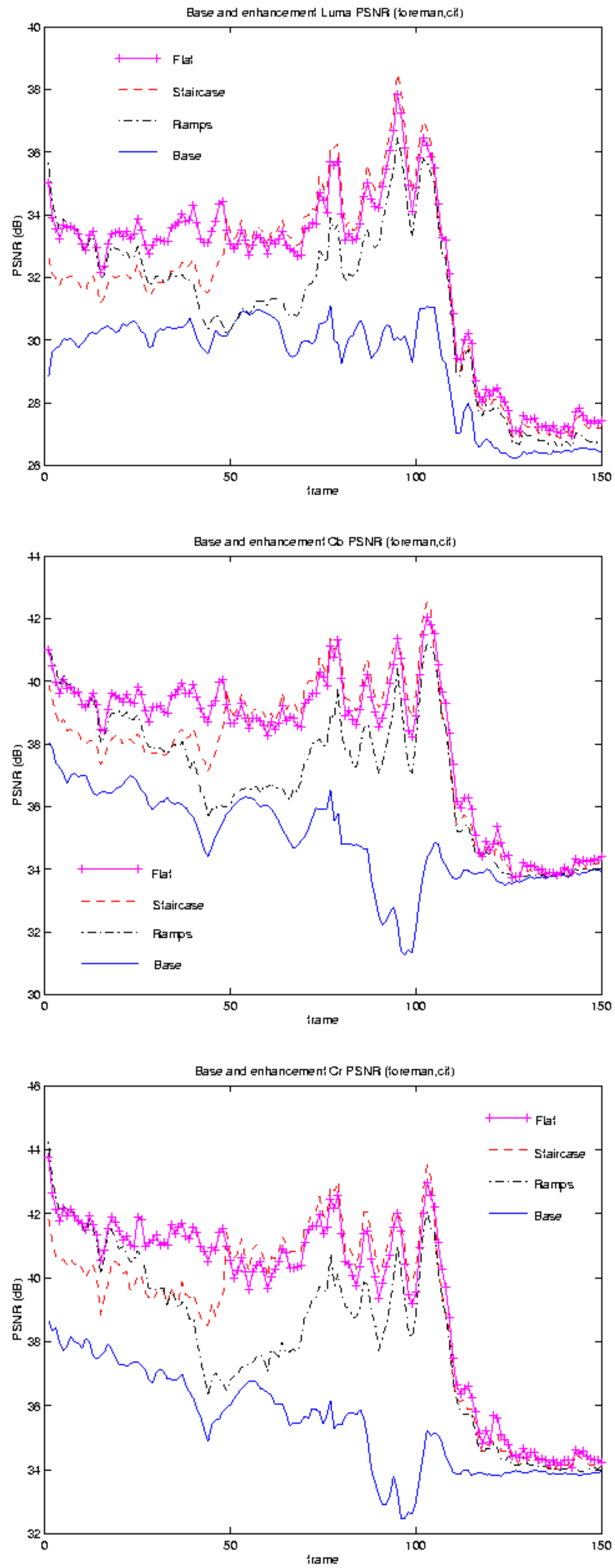

Figure 6 PSNR plots for Foreman, CIF (with mid-processing)

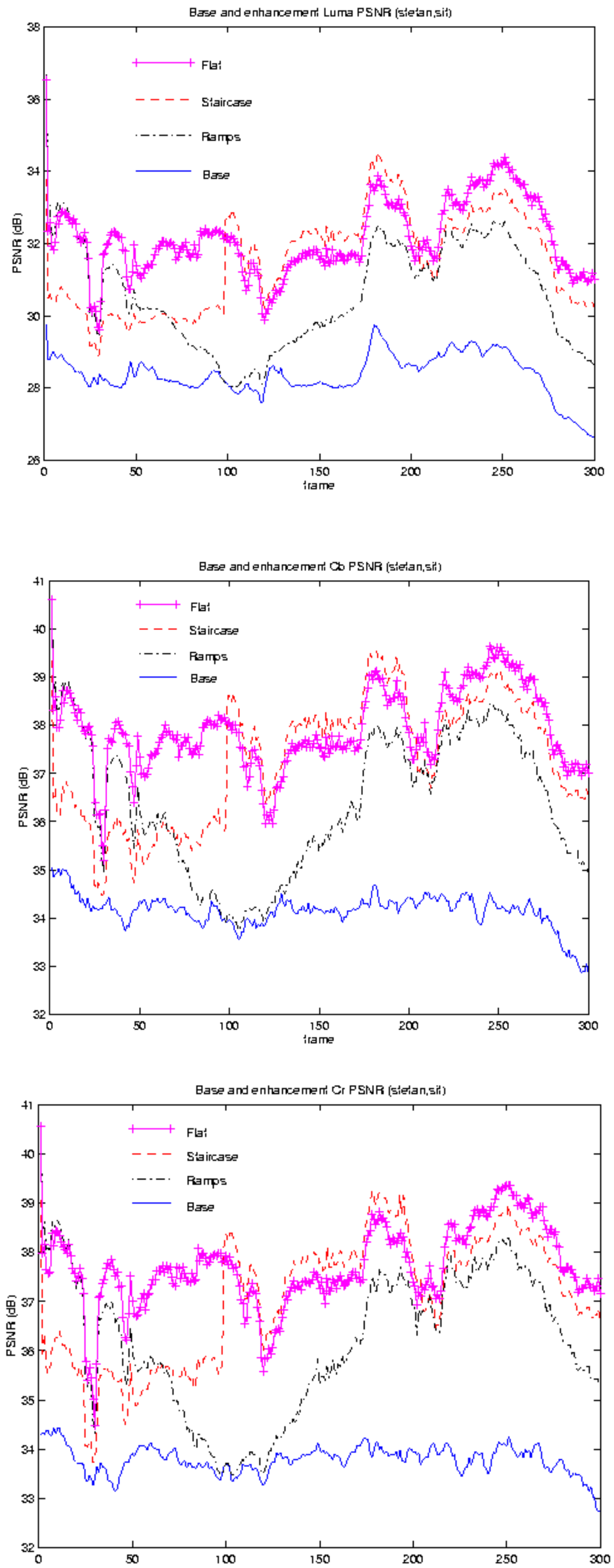

Figure 7 PSNR plots for Stefan, SIF (with mid-processing)