SONY
make.believe

Parametric Adaptive Loop Filter

Ehsan Maani, Wei Liu

Proposal
JCTVC-F285

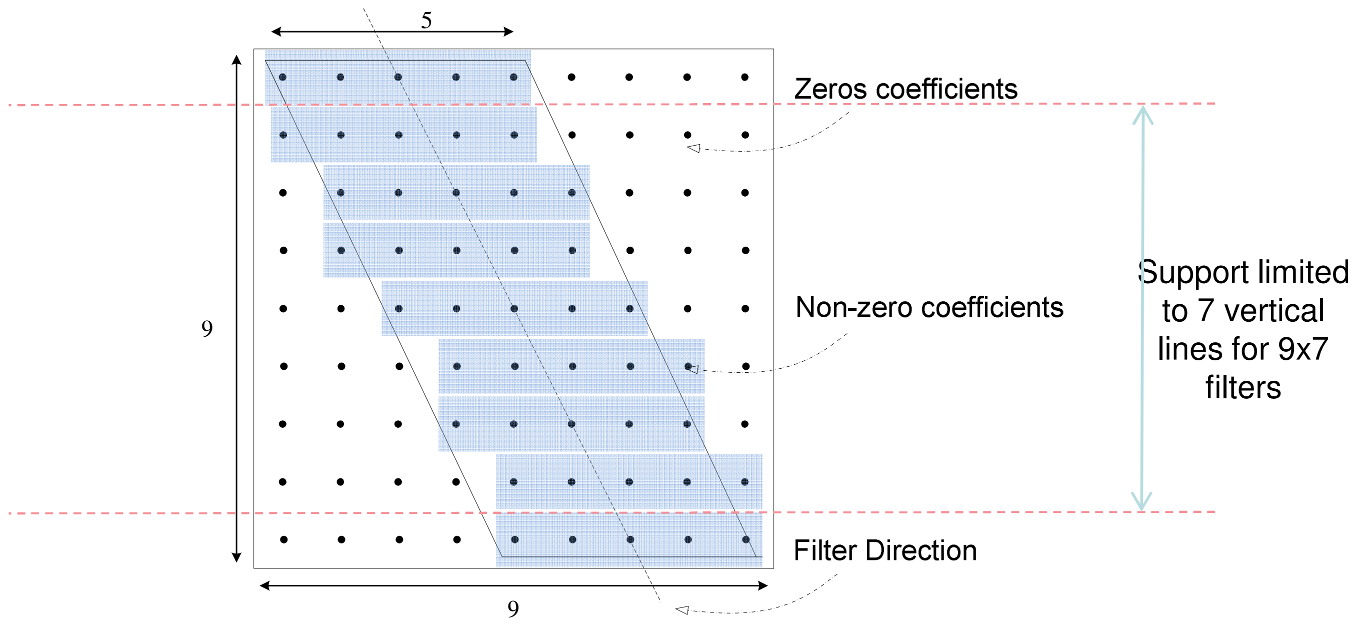
July 12 2011

Introduction

- PALF uses a set of fixed filters, identified by
  - direction and bandwidth (strength)
- Why fixed filters?
  - Lower decoder complexity (fixed coefficient multiplication can be optimized)
  - Lower encoder complexity (No need to derive filter coefficients)
  - Lower encoder delay (one LCU delay as opposed to one frame delay of ALF)
  - Higher coding gain when combined with ALF, mainly due to:
    - Online filter for all directions, strengths cannot usually be sent due to rate-constraint, however, if necessary, the index to a fixed filter can be transmitted efficiently.
      - ***Especially true at low bit-rates***
    - Local adaptively (more filters to choose from at local level)
    - Offline filters can be derived using more complex block-level/pixel level classifications (offline computation)

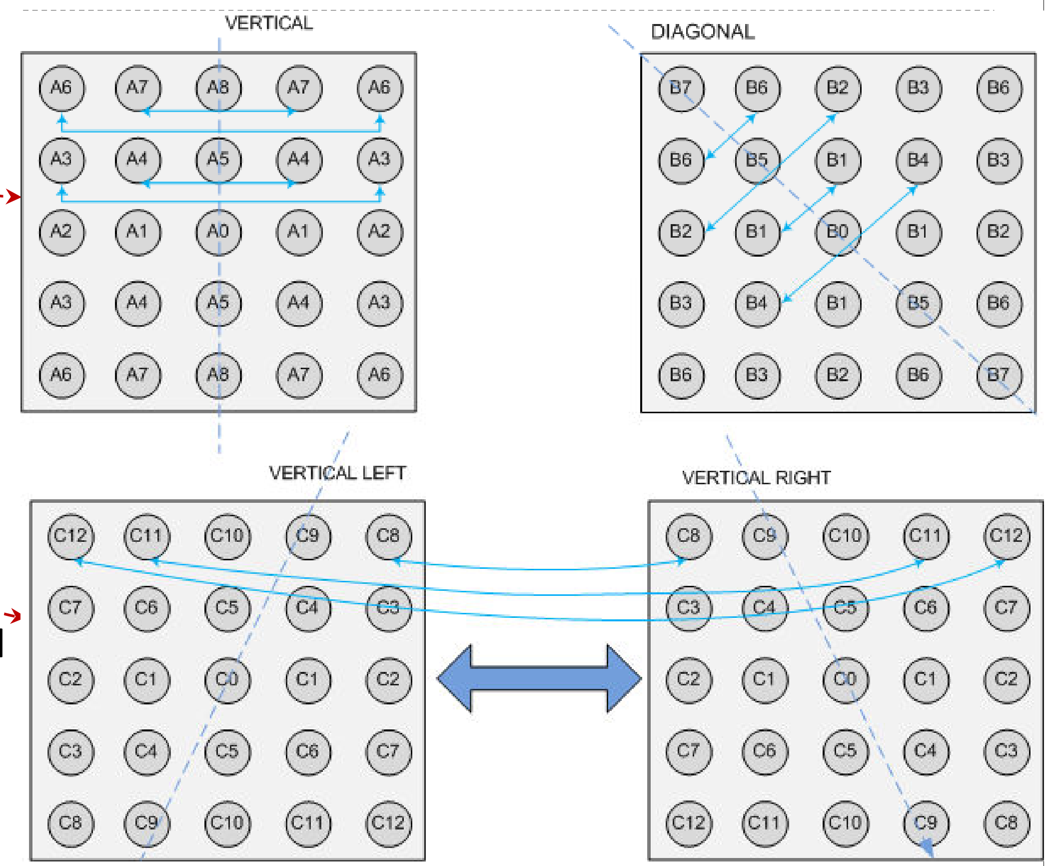
Filter Shape

- Non-zero coefficients are assumed to be along the direction
  - Support of the filter perpendicular to the direction is less than that along the direction
  - Example:



Filter Symmetry

- Mirror symmetry with respect to direction
  - Only Horizontal, Vertical, Diagonal
    - Reduces the number of coefficients
- Mirror symmetry with respect to vertical axis ***between filters***
  - Only for “vertical left”, “horizontal down”, “vertical right”, and “horizontal up”
    - Reduces the number of stored filter directions by 2



Filter Symmetry

- Rotational symmetry between filters w/ orthogonal directions
  - This symmetry only applies to 5x5, 7x7, 9x9 filter shapes (symmetric)
  - It applies to all directions
    - Reduces number of stored filters by ***half***

- Examples (5x5 chroma):

Horizontal-Down

-15	50	-41	12	0
19	-52	18	-100	39
-7	0	1177	0	-7
39	-100	18	-52	19
0	12	41	50	-15

Mirror symmetry w/ respect to vertical axis

Vertical

-2	-9	-34	-9	-2
8	4	147	4	8
-52	74	751	74	-52
8	4	147	4	8
-2	-9	-34	-9	-2

Horizontal

-2	8	-52	8	-2
-9	-4	74	4	-9
-34	147	751	147	-34
-9	-4	74	4	-9
-2	8	-52	8	-2

Horizontal-UP

0	12	-41	50	-15
39	-100	18	-52	19
-7	0	1177	0	-7
19	-52	18	-100	39
-15	50	41	12	0

Coeffs mirrored w/ respect direction of filter
Rotational symmetry

Filter Symmetry

- Considering these symmetries:
  - Total number of coefficients that need to be stored
    - 9x7 Luma: 244
    - 5x5 Chroma: 168

    - 9x5 Luma: 176
    - 5x3 Chroma: 200
      - For 5x3 rotational symmetry cannot be used as in 5x5. So, number of filters is twice as much

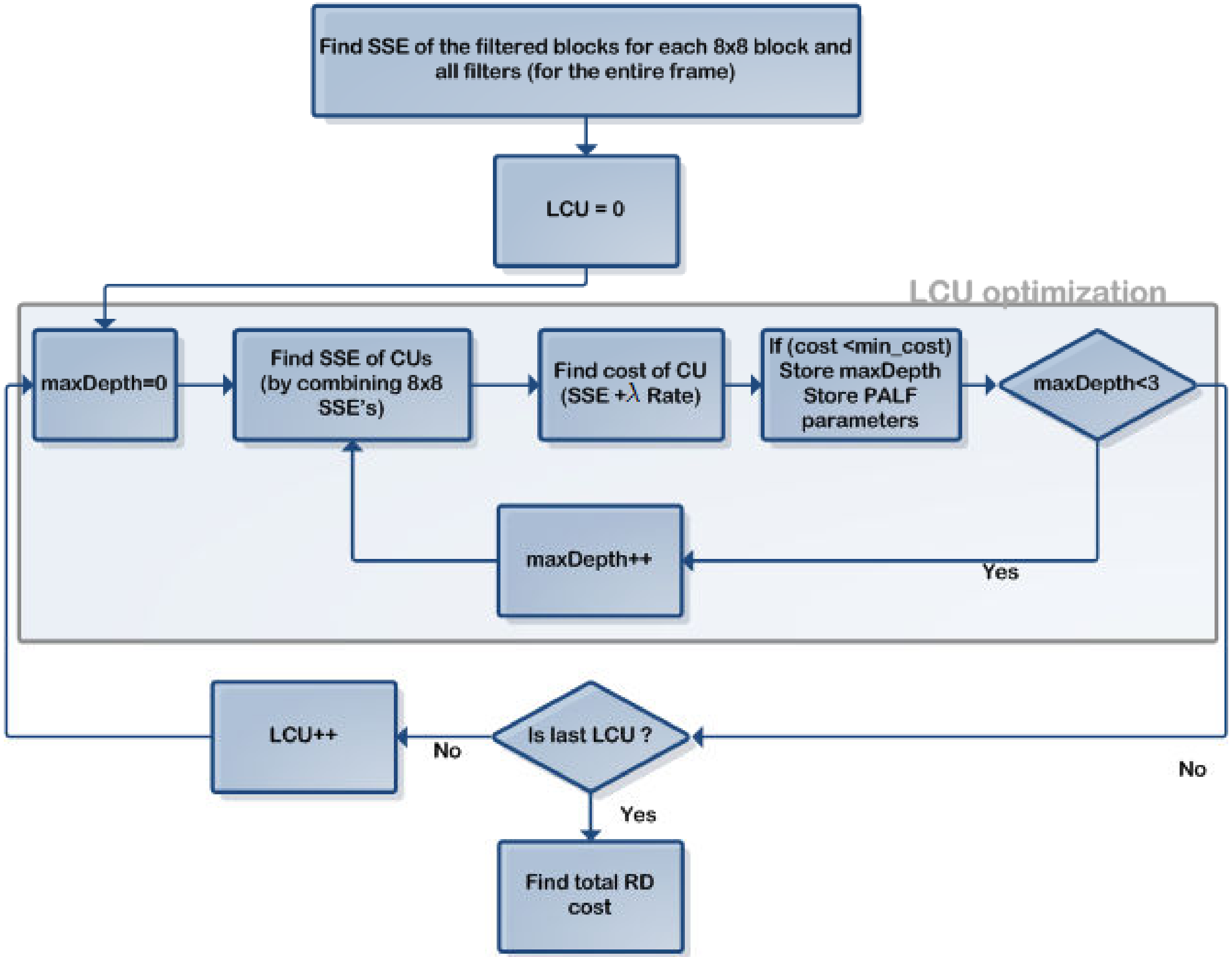
This is a THEORETICAL LIMIT: If according to the symmetry, a coefficient is needed more than one time in more than one filter, it is counted only ONCE.

Assuming an efficient implementation as shown in Appendix A.

Signaling (PALF ONLY)

- Parameter signaling
  - Control depth
    - For each LCU, a maxDepth is decided and signaled to the decoder.
    - Other control parameters are signaled only if CU depth is less than or equal to maxDepth
      - maxDepth allows encoder to complete the optimization of an LCU before going to next
  - Strength
    - For each CU with depth <= maxDepth, strength is transmitted
    - A prediction is made from neighboring CUs. Truncated unary codes are used.
    - Strength index is from the set {0, 1, 2, 3, 4}
      - Filter strength of “0” means no filtering.
  - Direction
    - If strength of CU is greater than 0, direction is also transmitted
    - A prediction is made from neighboring CU. If filter direction is the same as prediction 1 bit otherwise 4 bit is transmitted (fixed-length code for the remainder of directions)
    - Direction is from the set {0,1,2,3,4,5,6,7} which are the same as AVC Intra (w/o DC)

Encoder PALF Optimization



Combination with ALF

- The online filter is added as strength 1 to set of PALF filter strengths
  - So, the possible strength index set is {0, 1 ,2 ,3 ,4 ,5}
    - 0: no filtering, 1: online filter, and 2-5 are the four fixed filter strengths
  - Filter direction is not transmitted for filter strengths of 0 and 1
- The mask of online filter is set to 0 when strength is not 1
  - Only regions that use the online filter are included for the next round of training
- Slice-level combination modes (signaled to decoder):

0) ALF + without CU on/off

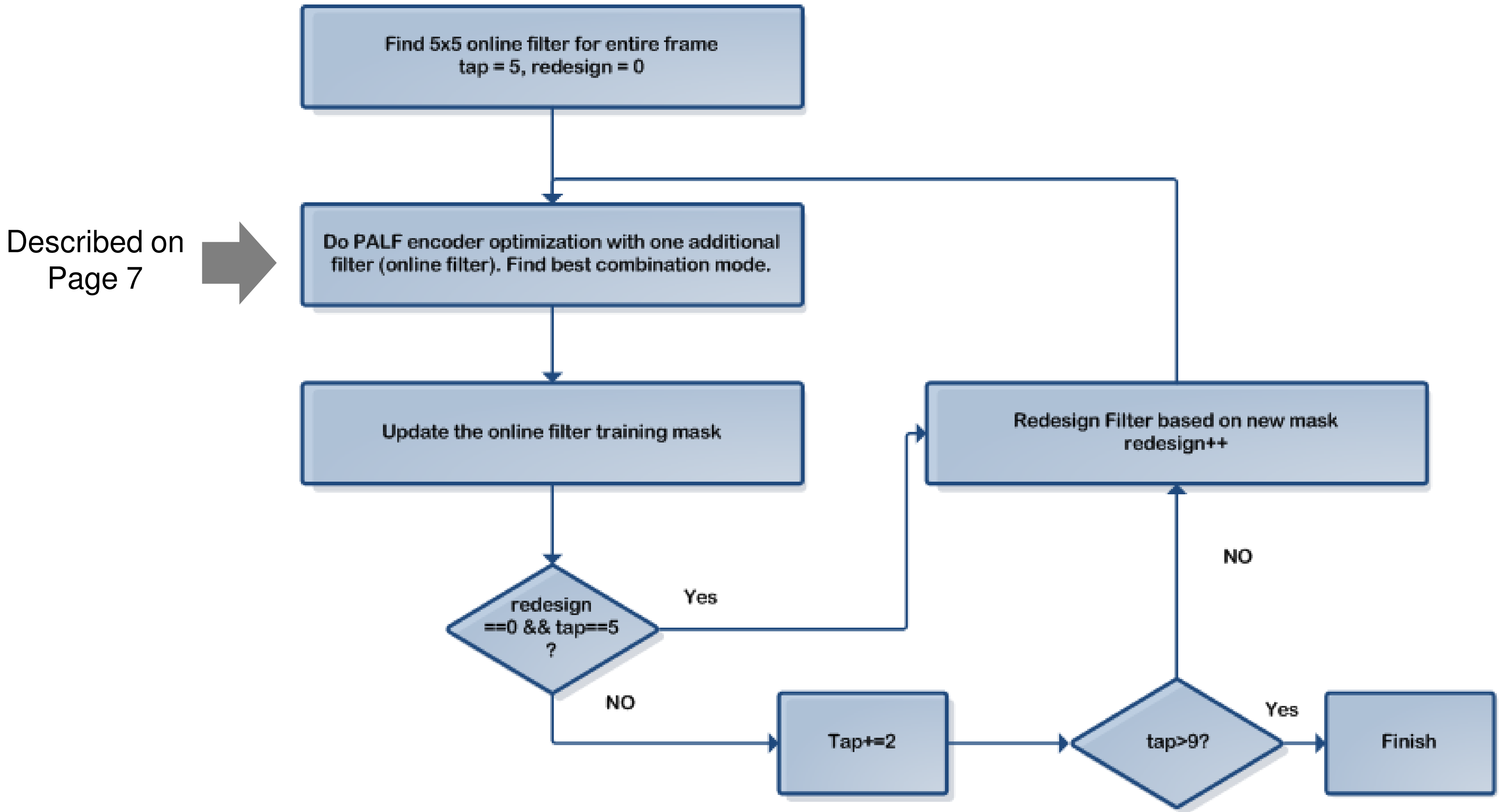
1) ALF + CU on/off (switching)

2) PALF + CU filter switching

3) ALF + PALF + CU filter switching

4) Same as 3, without the all-pass filter (saves signaling bits)

Hybrid ALF (Encoder)



Hybrid ALF (Decoder)

- There is minimal changes in the decoder
  - Decoder just applies the correct filter signaled by the filter strength
    - 0: no filter
    - 1: online filter (needs to decode filter coefficients and calculate imgVar)
    - 2-5: fixed filters

Results Set 1:

chroma_lambda =luma_lambda (HM3 default)

Luma: 9x7
# of possible filters to signal: 32
# of filters to store: 24

Chroma: 5x5
# of possible filters to signal: 32
# of filters to store: 12

	PALF+ALF			PALF			ALF		
	All intra HE			All intra LC			All intra LC		
	Y	U	V	Y	U	V	Y	U	V
Class A	0.2	-1.6	-2.8	-2.4	-1.3	-1.7	-3.8	-1.0	-1.4
Class B	0.1	-1.1	-2.2	-1.2	-2.2	-2.5	-2.2	-2.3	-1.7
Class C	-0.2	-0.6	-1.2	-1.3	-2.9	-3.9	-2.0	-2.6	-3.7
Class D	-0.2	0.1	-0.1	-0.7	-2.2	-2.6	-0.9	-2.2	-2.7
Class E	0.3	-1.9	-1.6	-1.9	-2.5	-3.6	-3.3	-3.8	-4.6
Overall	**0.0**	**-1.0**	**-1.6**	**-1.5**	**-2.2**	**-2.8**	**-2.4**	**-2.3**	**-2.7**
Enc Time[%]	99%			115%			139%		
Dec Time[%]	104%			125%			155%		
	Random access HE			Random access LC			Random access LC		
	Y	U	V	Y	U	V	Y	U	V
Class A	-0.1	-3.5	-4.9	-4.1	-4.6	-4.6	-6.2	-3.0	-3.9
Class B	-0.3	-3.2	-3.9	-2.6	-3.6	-2.8	-4.3	-1.4	0.0
Class C	-0.4	-2.5	-2.6	-2.5	-2.1	-3.1	-3.0	0.8	-0.6
Class D	-0.5	-0.8	-1.2	-2.8	-0.9	-0.9	-2.7	0.6	0.7
Class E									
Overall	**-0.3**	**-2.5**	**-3.2**	**-3.0**	**-2.8**	**-2.9**	**-4.1**	**-0.8**	**-0.9**
Enc Time[%]	100%			103%			107%		
Dec Time[%]	101%			116%			129%		
	Low delay (B) HE			Low delay (B) LC			Low delay (B) LC		
	Y	U	V	Y	U	V	Y	U	V
Class A									
Class B	-0.6	-4.8	-6.1	-3.5	-6.4	-6.3	-4.8	-1.1	-0.4
Class C	-0.8	-4.3	-5.3	-3.5	-4.6	-6.5	-3.8	0.7	-1.0
Class D	-0.8	-1.6	-2.1	-2.7	-2.0	-2.0	-2.5	0.4	0.6
Class E	-1.1	-5.1	-7.7	-5.2	-11.8	-11.5	-6.0	-6.6	-1.0
Overall	**-0.8**	**-3.9**	**-5.2**	**-3.6**	**-5.9**	**-6.2**	**-4.2**	**-1.3**	**-0.4**
Enc Time[%]	100%			103%			106%		
Dec Time[%]	101%			116%			131%		
	Low delay (P) HE			Low delay (P) LC			Low delay (P) LC		
	Y	U	V	Y	U	V	Y	U	V
Class A									
Class B	-0.7	-5.4	-6.5	-5.1	-8.4	-7.7	-6.9	-2.5	-1.5
Class C	-0.9	-4.4	-5.6	-3.3	-5.0	-6.7	-3.6	0.7	-1.0
Class D	-0.8	-1.9	-2.7	-1.9	-2.7	-2.7	-1.5	0.0	0.5
Class E	-1.3	-5.6	-8.6	-7.9	-14.3	-13.4	-8.7	-9.0	-2.4
Overall	**-0.9**	**-4.3**	**-5.7**	**-4.4**	**-7.2**	**-7.3**	**-5.1**	**-2.3**	**-1.1**
Enc Time[%]	100%			105%			110%		
Dec Time[%]	101%			119%			135%		

Results Set 2

chroma_lambda =1.8*luma_lambda

Luma: 9x7
# of possible filters to signal: 32
# of filters to store: 24

Chroma: 5x5
# of possible filters to signal: 32
# of filters to store: 12

	PALF+ALF			PALF		
	All intra HE			All intra LC		
	Y	U	V	Y	U	V
Class A	0.0	-0.6	-1.0	-2.5	-0.6	-1.0
Class B	-0.1	-0.5	-1.0	-1.3	-1.3	-1.5
Class C	-0.3	-0.2	-0.6	-1.4	-2.5	-3.4
Class D	-0.2	1.0	1.0	-0.7	-1.8	-2.1
Class E	0.1	-0.1	-0.2	-2.1	-0.2	-0.9
Overall	**-0.1**	**-0.1**	**-0.4**	**-1.6**	**-1.3**	**-1.8**
Enc Time[%]	100%			116%		
Dec Time[%]	102%			124%		
	Random access HE			Random access LC		
	Y	U	V	Y	U	V
Class A	-0.2	-1.5	-2.1	-4.2	-2.9	-3.0
Class B	-0.4	-1.6	-1.9	-2.7	-1.7	-1.2
Class C	-0.5	-1.3	-1.2	-2.6	-1.2	-2.0
Class D	-0.6	0.0	-0.2	-2.9	-0.2	-0.1
Class E						
Overall	**-0.4**	**-1.1**	**-1.4**	**-3.1**	**-1.5**	**-1.6**
Enc Time[%]	100%			103%		
Dec Time[%]	100%			116%		
	Low delay (B) HE			Low delay (B) LC		
	Y	U	V	Y	U	V
Class A						
Class B	-0.7	-2.7	-3.4	-3.6	-3.5	-3.4
Class C	-1.0	-3.0	-3.7	-3.6	-3.0	-4.5
Class D	-0.9	-0.6	-0.4	-2.8	-1.0	-0.9
Class E	-1.2	-2.4	-4.1	-5.5	-6.4	-6.5
Overall	**-0.9**	**-2.2**	**-2.8**	**-3.7**	**-3.3**	**-3.6**
Enc Time[%]	100%			103%		
Dec Time[%]	100%			115%		
	Low delay (P) HE			Low delay (P) LC		
	Y	U	V	Y	U	V
Class A						
Class B	-0.8	-3.2	-3.7	-5.2	-5.3	-4.6
Class C	-1.0	-3.1	-3.6	-3.5	-3.5	-5.0
Class D	-0.9	-1.1	-1.1	-2.0	-1.4	-1.4
Class E	-1.3	-2.8	-4.7	-8.2	-9.0	-8.6
Overall	**-1.0**	**-2.6**	**-3.2**	**-4.5**	**-4.6**	**-4.7**
Enc Time[%]	101%			104%		
Dec Time[%]	99%			119%		

Results Set 3

chroma_lambda =1.8*luma_lambda

Luma: 9x5
# of possible filters to signal: 32
# of filters to store: 24

Chroma: 5x3
# of possible filters to signal: 32
# of filters to store: 24

	PALF All Intra HE			PALF All Intra LC		
	Y	U	V	Y	U	V
Class A	1.6	-0.1	-0.2	-2.4	-0.8	-1.2
Class B	1.0	0.4	-1.0	-1.2	-0.9	-1.3
Class C	0.6	0.5	0.0	-1.3	-2.1	-3.0
Class D	0.2	0.9	0.9	-0.7	-1.5	-1.6
Class E	1.7	-0.1	1.1	-1.9	-0.4	-0.9
Overall	**1.0**	**0.3**	**0.1**	**-1.5**	**-1.2**	**-1.6**
Enc Time[%]	89%			113%		
Dec Time[%]	86%			119%		

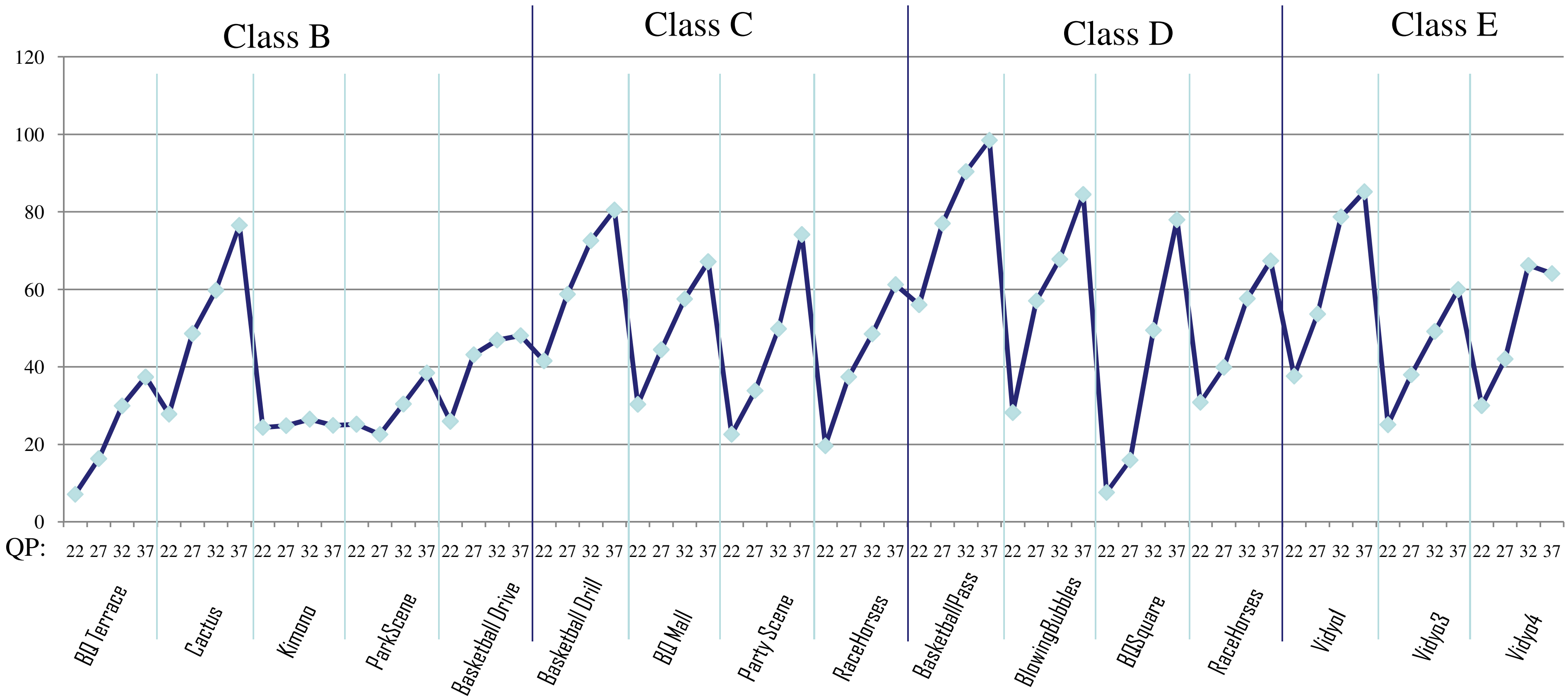
	Random Access HE			Random Access LC		
	Y	U	V	Y	U	V
Class A	2.2	1.4	0.8	-4.0	-3.0	-3.0
Class B	1.6	0.2	-1.3	-2.7	-1.6	-1.1
Class C	0.4	-1.3	-1.0	-2.5	-0.8	-1.7
Class D	0.1	-0.2	-0.4	-2.6	0.0	0.1
Class E						
Overall	**1.1**	**0.0**	**-0.5**	**-2.9**	**-1.4**	**-1.4**
Enc Time[%]	97%			102%		
Dec Time[%]	89%			110%		

	PALF Low delay P HE			PALF Low delay P LC		
	Y	U	V	Y	U	V
Class A						
Class B	1.9	-1.4	-3.0	-5.2	-4.9	-4.3
Class C	0.1	-3.6	-3.7	-3.5	-3.2	-4.5
Class D	-0.5	-1.0	-1.2	-2.0	-1.5	-1.3
Class E	0.8	-1.4	-4.8	-7.8	-7.5	-7.9
Overall	**0.6**	**-1.8**	**-3.1**	**-4.5**	**-4.1**	**-4.3**
Enc Time[%]	96%			103%		
Dec Time[%]	87%			111%		

	Low delay B HE			Low delay B LC		
	Y	U	V	Y	U	V
Class A						
Class B	1.1	-0.8	-2.3	-3.6	-3.0	-2.8
Class C	0.1	-3.2	-3.4	-3.5	-2.8	-4.2
Class D	-0.2	-0.7	-0.2	-2.6	-1.0	-1.0
Class E	0.6	-1.7	-3.1	-5.2	-5.3	-5.4
Overall	**0.4**	**-1.6**	**-2.2**	**-3.6**	**-2.9**	**-3.2**
Enc Time[%]	97%			102%		
Dec Time[%]	87%			110%		

Percentage of PALF CUs

Given a frame with PALF+ALF, percentage of PALF CUs on average:



Experiment Set 1. Low Delay (B). Luma component.

SONY

make.believe

Appendix: Example of storage and filtering

For illustration, here is an example of how number of filters is reduced. Consider 5x5 Chroma filter for:

- HORIZONTAL-DOWN:

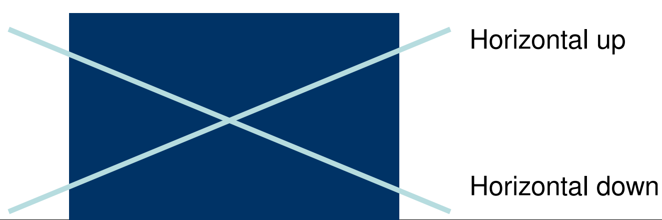
-15	50	-41	12	0
19	-52	18	-100	39
-7	0	1177	0	-7
39	-100	18	-52	19
0	12	41	50	-15

- HORIZONTAL-UP:

0	12	-41	50	-15
39	-100	18	-52	19
-7	0	1177	0	-7
19	-52	18	-100	39
-15	50	41	12	0

In memory (ROM) we only store one of them: piCoeff ={-15,50,-41,12,0,19,-52,18,-100,39,-7,0,1177 , 35}
THEN:

offset



SONY make.believe

```
for (y = 0; y < iHeight; y++)
{
    for (x = 0; x < iWidth; x++)
    {
        iSum = piCoeff[13];  qLoc = pLoc = pDec + x;
        iSum +=  (*pLoc++           ) *  (piCoeff[12]); qLoc--;
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[11]); iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[10]);
        pLoc += iDecStride - 5; qLoc -= iDecStride - 5;
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 9]);
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 8]); iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 7]);
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 6]); iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 5]);
        pLoc += iDecStride - 5; qLoc -= iDecStride - 5;
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 4]);
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 3]); iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 2]);
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 1]); iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 0]);
        iSum = (iSum + PALF_ROUND_OFFSET)>>PALF_NUM_BIT_SHIFT;
        pRest[x] = (Pel) Clip(iSum);
    }
    pRest += iRestStride;   pDec  += iDecStride;
}
```

Filtering for Horizontal Down

ONE filter (array) to filter 2 directions (using symmetry of Horizontal-Down and Horizontal-Up)

```
for (y = 0; y < iHeight; y++)
{
    for (x = 0; x < iWidth; x++)
    {
        iSum = piCoeff[13];  qLoc = pLoc = pDec + x;
        iSum +=  (*pLoc++           ) *  (piCoeff[12]); qLoc--;
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[11]); iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[10]);
        pLoc += iDecStride - 5; qLoc -= iDecStride - 5;
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 5]);
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 6]); iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 7]);
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 8]); iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 9]);
        pLoc += iDecStride - 5; qLoc -= iDecStride - 5;
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 0]);
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 1]); iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 2]);
        iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 3]); iSum +=  (*pLoc++ + *qLoc--) *  (piCoeff[ 4]);
        iSum = (iSum + PALF_ROUND_OFFSET)>>PALF_NUM_BIT_SHIFT;
        pRest[x] = (Pel) Clip(iSum);
    }
    pRest += iRestStride;   pDec  += iDecStride;
}
```

Filtering for Horizontal up